Refine Search

Search Results -

Terms	Documents
ep-704810$.did.	2

Database:
US Pre-Grant Publication Full-Text Database
US Patents Full-Text Database
US OCR Full-Text Database
EPO Abstracts Database
JPO Abstracts Database
Derwent World Patents Index
IBM Technical Disclosure Bulletins

Search:

Refine Search

Recall Text Clear Interrupt

Search History

DATE: Monday, December 31, 2007 **Purge Queries** Printable Copy Create Case

Set Name side by side	Query	Hit Count	Set Name result set
DB=PGPB,USPT,USOC,EPAB,JPAB,DWPI,TDBD; PLUR=YES; OP=OR			
L60	ep-704810$.did.	2	L60
L59	ep-717346$.did.	2	L59
L58	"ohtani, noriko".in.	33	L58
L57	"shibata, shogo".in.	85	L57
L56	"ueda, takaya".in.	22	L56
L55	"ikeda, yuji".in.	580	L55
L54	"fumiaki, ito".in.	0	L54
L53	"ito, fumiaki".in.	69	L53
L52	"fumiakito, ito".in.	0	L52
L51	707/104.1	9758	L51
L50	707/1	10584	L50
L49	370/389	12792	L49
L48	370.clas.	118990	L48

L47	715/532	261	L47
L46	715/531	752	L46
L45	715/530	1698	L45
L44	715/500	1814	L44
L43	715/513	4020	L43
L42	715/506	248	L42
L41	715.clas.	32337	L41
L40	707.clas.	64598	L40
L39	707/532	507	L39
L38	707/531	983	L38
L37	707/530	1434	L37
L36	707/506	311	L36
L35	707/513	3092	L35
L34	707/500	1769	L34
L33	707/200	6723	L33
L32	707/100	11355	L32
L31	707/8	3286	L31
L30	707/5	5587	L30
L29	707/3	11892	L29
L28	707/1	10584	L28
DB=PGPB,USPT,USOC; PLUR=YES; OP=OR			
L27	l4 and l5	423	L27
DB=EPAB,JPAB; PLUR=YES; OP=OR			
L26	l4 and l5	0	L26
DB=DWPI,TDBD; PLUR=YES; OP=OR			
L25	l4 and l5	0	L25
L24	l6 and l7	0	L24
DB=JPAB; PLUR=YES; OP=OR			
L23	l6 and l7	0	L23
DB=EPAB,JPAB; PLUR=YES; OP=OR			
L22	l6 and l7	0	L22
DB=PGPB,USPT,USOC; PLUR=YES; OP=OR			
L21	l6 and l7	410	L21
DB=PGPB,USPT,USOC,EPAB,JPAB,DWPI,TDBD; PLUR=YES; OP=OR			
L20	l6 and L17	4	L20
L19	l7 and L17	598	L19
L18	l8 and L17	4	L18
L17	709/209	703	L17
L16	l8 and L13	80	L16
L15	l8 and L11	63	L15
L14	l7 and L13	29622	L14

L13	715.clas.	32337	L13
L12	l7 and L11	54776	L12
L11	709.clas.	60270	L11
L10	l8 and L9	208	L10
L9	707.clas.	64598	L9
L8	l6 and L7	410	L8
L7	(rank$2 or weight$2 or scoring or ranking or weighting or scor$2 or comput$2 or compute)	5433126	L7
L6	l4 and L5	423	L6
L5	(candidate or primary or main or first) and (folder or retainer or container)	1023629	L5
L4	L3 and (search or search$2)	436	L4
L3	l1 and l2	848	L3
L2	(nested with folders or nested near folders or nested adj folders or inside near folders or inside adj folders or inside with folders)	2151	L2
L1	(folder or retainer or container) and document	91896	L1

END OF SEARCH HISTORY

Refine Search

Search Results -

Terms	Documents
ep-704810$.did.	2

Database: US Pre-Grant Publication Full-Text Database; US Patents Full-Text Database; US OCR Full-Text Database; EPO Abstracts Database; JPO Abstracts Database; Derwent World Patents Index; IBM Technical Disclosure Bulletins

Search:

Refine Search | Recall Text | Clear | Interrupt

Search History

DATE: Monday, December 31, 2007 **Purge Queries** Printable Copy Create Case

Set Name side by side	Query	Hit Count	Set Name result set
DB=PGPB,USPT,USOC,EPAB,JPAB,DWPI,TDBD; PLUR=YES; OP=OR			
L60	ep-704810$.did.	2	L60
L59	ep-717346$.did.	2	L59
L58	"ohtani, noriko".in.	33	L58
L57	"shibata, shogo".in.	85	L57
L56	"ueda, takaya".in.	22	L56
L55	"ikeda, yuji".in.	580	L55
L54	"fumiaki, ito".in.	0	L54
L53	"ito, fumiaki".in.	69	L53
L52	"fumiakito, ito".in.	0	L52
L51	707/104.1	9758	L51
L50	707/1	10584	L50
L49	370/389	12792	L49
L48	370.clas.	118990	L48

L47	715/532	261	L47
L46	715/531	752	L46
L45	715/530	1698	L45
L44	715/500	1814	L44
L43	715/513	4020	L43
L42	715/506	248	L42
L41	715.clas.	32337	L41
L40	707.clas.	64598	L40
L39	707/532	507	L39
L38	707/531	983	L38
L37	707/530	1434	L37
L36	707/506	311	L36
L35	707/513	3092	L35
L34	707/500	1769	L34
L33	707/200	6723	L33
L32	707/100	11355	L32
L31	707/8	3286	L31
L30	707/5	5587	L30
L29	707/3	11892	L29
L28	707/1	10584	L28
DB=PGPB,USPT,USOC; PLUR=YES; OP=OR			
L27	l4 and l5	423	L27
DB=EPAB,JPAB; PLUR=YES; OP=OR			
L26	l4 and l5	0	L26
DB=DWPI,TDBD; PLUR=YES; OP=OR			
L25	l4 and l5	0	L25
L24	l6 and l7	0	L24
DB=JPAB; PLUR=YES; OP=OR			
L23	l6 and l7	0	L23
DB=EPAB,JPAB; PLUR=YES; OP=OR			
L22	l6 and l7	0	L22
DB=PGPB,USPT,USOC; PLUR=YES; OP=OR			
L21	l6 and l7	410	L21
DB=PGPB,USPT,USOC,EPAB,JPAB,DWPI,TDBD; PLUR=YES; OP=OR			
L20	l6 and L17	4	L20
L19	l7 and L17	598	L19
L18	l8 and L17	4	L18
L17	709/209	703	L17
L16	l8 and L13	80	L16
L15	l8 and L11	63	L15
L14	l7 and L13	29622	L14

L13	715.clas.	32337	L13
L12	l7 and L11	54776	L12
L11	709.clas.	60270	L11
L10	l8 and L9	208	L10
L9	707.clas.	64598	L9
L8	l6 and L7	410	L8
L7	(rank$2 or weight$2 or scoring or ranking or weighting or scor$2 or comput$2 or compute)	5433126	L7
L6	l4 and L5	423	L6
L5	(candidate or primary or main or first) and (folder or retainer or container)	1023629	L5
L4	L3 and (search or search$2)	436	L4
L3	l1 and l2	848	L3
L2	(nested with folders or nested near folders or nested adj folders or inside near folders or inside adj folders or inside with folders)	2151	L2
L1	(folder or retainer or container) and document	91896	L1

END OF SEARCH HISTORY

Refine Search

Search Results -

Terms	Documents
5884321.pn.	2

Database:
US Pre-Grant Publication Full-Text Database
US Patents Full-Text Database
US OCR Full-Text Database
EPO Abstracts Database
JPO Abstracts Database
Derwent World Patents Index
IBM Technical Disclosure Bulletins

Search:

Refine Search

Recall Text Clear Interrupt

Search History

DATE: Monday, December 31, 2007 **Purge Queries** Printable Copy Create Case

Set Name side by side	Query	Hit Count	Set Name result set
	DB=PGPB,USPT,USOC,EPAB,JPAB,DWPI,TDBD; PLUR=YES; OP=OR		
L16	5884321.pn.	2	L16
L15	5870711.pn.	2	L15
L14	5280609.pn.	2	L14
	DB=USPT; PLUR=YES; OP=OR		
L13	'4876665'.pn.	1	L13
L12	'4933848'.pn.	1	L12
L11	'5101345'.pn.	1	L11
L10	'5101345'.pn.	1	L10
	DB=PGPB,USPT,USOC,EPAB,JPAB,DWPI,TDBD; PLUR=YES; OP=OR		
L9	5255389.pn.	2	L9
	DB=USPT; PLUR=YES; OP=OR		
L8	("5832470")[URPN]	28	L8
	DB=PGPB,USPT,USOC,EPAB,JPAB,DWPI,TDBD; PLUR=YES; OP=OR		

L7	l5 and (folder or folders)	1	L7
L6	5832470.pn.	2	L6
DB=USPT; PLUR=YES; OP=OR			
L5	(5428727 | 4958284 | 5355497 | 5463773 | 4760606 | 4719571 | 5581752 | 5168533 | 5590317 | 5508912 | 5367672 | 5201047 | 5519865)![PN]	13	L5
L4	("5832470")[PN]	1	L4
DB=PGPB,USPT,USOC,EPAB,JPAB,DWPI,TDBD; PLUR=YES; OP=OR			
L3	5832470.pn.	2	L3
L2	ep-270360$.did.	2	L2
L1	ep-509945$.did.	2	L1

END OF SEARCH HISTORY

Freeform Search

Database: US Pre-Grant Publication Full-Text Database
US Patents Full-Text Database
US OCR Full-Text Database
EPO Abstracts Database
JPO Abstracts Database
Derwent World Patents Index
IBM Technical Disclosure Bulletins

Term:

Display: 10 **Documents in Display Format:** TI **Starting with Number** 1

Generate: ○ **Hit List** ◉ **Hit Count** ○ **Side by Side** ○ **Image**

Search Clear Interrupt

Search History

DATE: Monday, December 31, 2007 **Purge Queries** Printable Copy Create Case

Set Name side by side	Query	Hit Count	Set Name result set
DB=USPT; PLUR=YES; OP=OR			
L7	("5418946")[URPN]	52	L7
L6	(4601021 | 4999790 | 4653021 | 5299303 | 5060277 | 4319337 | 5295062 | 4468728 | 5247666)![PN]	9	L6
L5	("5418946")[PN]	1	L5
DB=PGPB,USPT,USOC,EPAB,JPAB,DWPI,TDBD; PLUR=YES; OP=OR			
L4	5418946.pn.	2	L4
L3	6363400.pn.	2	L3
L2	5812995.pn.	2	L2
L1	5418946.pn.	2	L1

END OF SEARCH HISTORY

Terms used: **categorizing e mail folders** Found **120** of **216,199**

Sort results by: relevance
Display results: expanded form
Save results to a Binder
Search Tips
Open results in a new window

Try an Advanced Search
Try this search in The ACM Guide

Results 1 - 20 of 120 Result page: **1** 2 3 4 5 6 7 next

Relevance scale

1 Late breaking result papers: Collections: flexible, essential tools for information management

David R. Karger, Dennis Quan

April 2004 **CHI '04 extended abstracts on Human factors in computing systems CHI '04**

Publisher: ACM Press

Full text available: pdf(255.14 KB) Additional Information: full citation, abstract, references, index terms

While collections-aggregation mechanisms such as folders, buddy lists, photo albums, etc.-clearly play a central role in information management, the potential benefits of true first class support for collections are masked by disparate implementations that force users to pay attention to technological distinctions such as application, format, and protocol. We argue that systems should expose a single unified concept of collection and that concepts such as portals, cross-application projects, cus ...

Keywords: collections, folders, portals, semistructured data, taxonomy

2 An experimental comparison of naive Bayesian and keyword-based anti-spam filtering with personal e-mail messages

Ion Androutsopoulos, John Koutsias, Konstantinos V. Chandrinos, Constantine D. Spyropoulos

July 2000 **Proceedings of the 23rd annual international ACM SIGIR conference on Research and development in information retrieval SIGIR '00**

Publisher: ACM Press

Full text available: pdf(811.41 KB) Additional Information: full citation, abstract, references, citings, index terms

The growing problem of unsolicited bulk e-mail, also known as "spam", has generated a need for reliable anti-spam e-mail filters. Filters of this type have so far been based mostly on manually constructed keyword patterns. An alternative approach has recently been proposed, whereby a Naive Bayesian classifier is trained automatically to detect spam messages. We test this approach on a large collection of personal e-mail messages, which we make publicly available in "encrypte ...

Keywords: evaluation (general), filtering&slash;routing, machine learning and IR, test collections, text categorization

3 Grassroots: providing a uniform framework for communicating, sharing information, and organizing people

Kenichi Kamiya, Martin Röscheisen, Terry Winograd

April 1996 **Conference companion on Human factors in computing systems: common ground CHI '96**

Publisher: ACM Press

Full text available: pdf(315.96 KB) Additional Information: full citation, references, citings, index terms

4 Posters: Detecting action-items in e-mail

Paul N. Bennett, Jaime Carbonell

August 2005 **Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval SIGIR '05**

Publisher: ACM Press

Full text available: pdf(107.02 KB) Additional Information: full citation, references, index terms

Keywords: *n*-grams, SVMs, e-mail, text classification

5 Text mining as integration of several related research areas: report on KDD's workshop on text mining 2000

Marko Grobelnik, Dunja Mladenic, Natasa Milic-Frayling

December 2000 **ACM SIGKDD Explorations Newsletter**, Volume 2 Issue 2

Publisher: ACM Press

Full text available: pdf(381.17 KB) Additional Information: full citation, index terms

Keywords: KDD workshop report, information extraction, information retrieval, natural language processing, text mining

6 Gender difference trends in computer literacy of first-year students

Mark E. Hoffman, David R. Vance

March 2007 **ACM SIGCSE Bulletin , Proceedings of the 38th SIGCSE technical symposium on Computer science education SIGCSE '07**, Volume 39 Issue 1

Publisher: ACM Press

Full text available: pdf(210.01 KB) Additional Information: full citation, abstract, references, index terms

We administered a computer literacy survey of our incoming, first-year students for the past three years. Our purpose was not to measure application skill levels, but to understand students' perception of their own skills, to identify from whom they learned how to perform a set of technology tasks, and to understand how access to different Internet connection types affects perception and the sources of student technology learning. Over the years, fale, first-year students have increased to parit ...

Keywords: computer literacy, first-year students, from whom learned, gender, skill level, technology tasks

7 On integrating catalogs

Rakesh Agrawal, Ramakrishnan Srikant

April 2001 **Proceedings of the 10th international conference on World Wide Web**

WWW '01
Publisher: ACM Press
Full text available: pdf(231.96 KB) Additional Information: full citation, references, citings, index terms

Keywords: Web marketplaces, Web portals, catalog integration, categorization, classification, data mining

8 Research track posters: Single-pass online learning: performance, voting schemes and online feature selection
Vitor R. Carvalho, William W. Cohen
August 2006 **Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining KDD '06**
Publisher: ACM Press
Full text available: pdf(742.44 KB) Additional Information: full citation, abstract, references, index terms

To learn concepts over massive data streams, it is essential to design inference and learning methods that operate in real time with limited memory. Online learning methods such as perceptron or Winnow are naturally suited to stream processing; however, in practice multiple passes over the same training data are required to achieve accuracy comparable to state-of-the-art batch learners. In the current work we address the problem of training an on-line learner with a single passover the data. We ...

Keywords: averaging, online learning, voting, winnow

9 Industrial session: Task-based process know-how reuse and proactive information delivery in TaskNavigator
Harald Holz, Oleg Rostanin, Andreas Dengel, Takeshi Suzuki, Kaoru Maeda, Katsumi Kanasaki
November 2006 **Proceedings of the 15th ACM international conference on Information and knowledge management CIKM '06**
Publisher: ACM Press
Full text available: pdf(1.24 MB) Additional Information: full citation, abstract, references, index terms

Knowledge management approaches for weakly-structured, adhoc knowledge work processes need to be lightweight, i.e., they cannot rely on high upfront modeling efforts. This paper presents TaskNavigator, a novel prototype to support weakly-structured processes by integrating a standard task list application with a state-of-the-art document classification system. The resulting system allows for a task-oriented view on office workers' personal knowledge spaces in order to realize a proactive and con ...

Keywords: agile workflows, proactive information delivery, process-oriented knowledge management

10 Somewhere over the network: providing services for remote users at NPAC
Betty LaPlante
August 1990 **Proceedings of the 18th annual ACM SIGUCCS conference on User services SIGUCCS '90**
Publisher: ACM Press
Full text available: pdf(441.15 KB) Additional Information: full citation, abstract, index terms

Is your institution going to establish a regional or statewide high performance computing center? Are you supporting users in remote locations who, therefore, cannot simply walk

into your center for help? Such a situation presents unique challenges demanding innovative solutions. The Northeast Parallel Architectures Center (NPAC) at Syracuse University has been handling these challenges since it opened in the spring of 1987. This paper will describe some of the successful tools and strategi ...

11 Adaptivity in speech-based multilingual e-mail client

Esa-Pekka Salonen, Mikko Hartikainen, Markku Turunen, Jaakko Hakulinen, Jyrki Rissanen, Kari Kanto, Kristiina Jokinen

October 2004 **Proceedings of the third Nordic conference on Human-computer interaction NordiCHI '04**

Publisher: ACM Press

Full text available: pdf(239.07 KB) Additional Information: full citation, abstract, references, index terms

In speech interfaces users must be aware what can be done with the system - in other words, the system must provide information to help the users to know what to say. We have addressed this challenge by using adaptive techniques that support the learning and use of speech applications. We describe how adaptivity can be supported on architectural level, how user modeling can help to make the interface more adaptive, how integrated tutoring teaches the users to use speech applications and how c ...

Keywords: adaptation, voice I/O

12 Media: Personal vs. commercial content: the similarities between consumer use of photos and music

Frank Bentley, Crysta Metcalf, Gunnar Harboe

April 2006 **Proceedings of the SIGCHI conference on Human Factors in computing systems CHI '06**

Publisher: ACM Press

Full text available: pdf(469.83 KB) Additional Information: full citation, abstract, references, index terms

We describe the results of two ethnographic-style studies that investigated consumer use of photos and music respectively. Although the studies were designed, executed, and analyzed separately, in our findings we discovered striking similarities between the ways in which our participants used personally captured photos and commercially purchased music. These findings have implications for the design of future systems with respect to handling and sharing content in photo or music form. We discuss ...

Keywords: consumer, ethnographic study, music, photo, sharing

13 Semantic interfaces and OWL tools: How to make a semantic web browser

D. A. Quan, R. Karger

May 2004 **Proceedings of the 13th international conference on World Wide Web WWW '04**

Publisher: ACM Press

Full text available: pdf(484.00 KB) Additional Information: full citation, abstract, references, citings, index terms

Two important architectural choices underlie the success of the Web: numerous, independently operated servers speak a common protocol, and a single type of client the Web browser provides point-and-click access to the content and services on these decentralized servers. However, because HTML marries content and presentation into a single representation, end users are often stuck with inappropriate choices made by the Web site designer of how to work with and view the content. RDF metadata on the ...

Keywords: bioinformatics, rdf, semantic web, user interface, web services

14 Email overload: exploring personal information management of email

Steve Whittaker, Candace Sidner

April 1996 **Proceedings of the SIGCHI conference on Human factors in computing systems: common ground CHI '96**

Publisher: ACM Press

Full text available: pdf(1.40 MB) html(50.38 KB) Additional Information: full citation, references, citings, index terms

Keywords: asynchronous communication, email, empirical studies, ethnography, filing, information overload, interpersonal communication, personal information management, task management

15 MailCat: an intelligent assistant for organizing e-mail

Richard B. Segal, Jeffrey O. Kephart

April 1999 **Proceedings of the third annual conference on Autonomous Agents AGENTS '99**

Publisher: ACM Press

Full text available: pdf(862.63 KB) Additional Information: full citation, references, citings, index terms

16 Improving the browsing experience: Information search and re-access strategies of experienced web users

Anne Aula, Natalie Jhaveri, Mika Käki

May 2005 **Proceedings of the 14th international conference on World Wide Web WWW '05**

Publisher: ACM Press

Full text available: pdf(212.45 KB) Additional Information: full citation, abstract, references, citings, index terms

Experienced web users have strategies for information search and re-access that are not directly supported by web browsers or search engines. We studied how prevalent these strategies are and whether even experienced users have problems with searching and re-accessing information. With this aim, we conducted a survey with 236 experienced web users. The results showed that this group has frequently used key strategies (e.g., using several browser windows in parallel) that they find important, whe ...

Keywords: experienced web users, information re-access, questionnaire study, web search

17 Where did you put it? Issues in the design and use of a group memory

Lucy M. Berlin, Robin Jeffries, Vicki L. O'Day, Andreas Paepcke, Cathleen Wharton

May 1993 **Proceedings of the INTERACT '93 and CHI '93 conference on Human factors in computing systems CHI '93**

Publisher: ACM Press

Full text available: pdf(1.07 MB) Additional Information: full citation, abstract, references, cited by, index terms

Collaborating teams of knowledge workers need a common repository in which to share information gathered by individuals or developed by the team. This is difficult to achieve in practice, because individual information access strategies break down with group

information—people can generally find things that are on their own messy desks and file systems, but not on other people's. The design challenge in a group memory is thus to enable low-effort informatio ...

Keywords: collaborative work, group conventions, group memory, information search and retrieval, information sharing

18 Genre, task, topic and time: facets of personal digital document management

Sarah Henderson

July 2005 **Proceedings of the 6th ACM SIGCHI New Zealand chapter's international conference on Computer-human interaction: making CHI natural CHINZ '05**

Publisher: ACM Press

Full text available: pdf(175.54 KB) Additional Information: full citation, abstract, references, index terms

Most operating systems provide the ability to create folders to contain documents, and to nest these folders to create a hierarchical organization. However, little is known about the kinds of folders people create using this type of organizing scheme, or how they structure those folders.Exploratory research was conducted, analyzing the folder structures of six knowledge workers and it was found that most folder names represent the genre, task, topic or time dimension of the documents they contai ...

Keywords: document organization, personal digital document management, personal information management

19 An experimental framework for email categorization and management

Kenricj Mock

September 2001 **Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval SIGIR '01**

Publisher: ACM Press

Full text available: pdf(174.00 KB) Additional Information: full citation, abstract, references, citings, index terms

Many problems are difficult to adequately explore until a prototype exists in order to elicit user feedback. One such problem is a system that automatically categorizes and manages email. Due to a myriad of user interface issues, a prototype is necessary to determine what techniques and technologies are effective in the email domain. This paper describes the implementation of an add-in for Microsoft Outlook 2000 TM that intends to address two problems with email: 1) help manage the inbo ...

Keywords: classification, email management, foltering

20 Information access in complex, poorly structured information spaces

Gerhard Fischer, Curt Stevens

March 1991 **Proceedings of the SIGCHI conference on Human factors in computing systems: Reaching through technology CHI '91**

Publisher: ACM Press

Full text available: pdf(794.73 KB) Additional Information: full citation, references, citings, index terms

Results for **"((email folders)<in>metadata)"**
Your search matched **2** of **1715275** documents.
A maximum of **100** results are displayed, **25** to a page, sorted by **Relevance** in **Descending** order.

☐ 1. **eMailSift: eMail classification based on structure and content**
Aery, M.; Chakravarthy, S.;
Data Mining, Fifth IEEE International Conference on
27-30 Nov. 2005 Page(s):8 pp.
Digital Object Identifier 10.1109/ICDM.2005.58
AbstractPlus | Full Text: PDF(160 KB) IEEE CNF
Rights and Permissions

☐ 2. **Self-maintained folder hierarchies as document repositories**
Eder, J.; Krumpholz, A.; Biliris, A.; Panagos, E.;
Digital Libraries: Research and Practice, 2000 Kyoto, International Conferenc
13-16 Nov. 2000 Page(s):400 - 407
Digital Object Identifier 10.1109/DLRP.2000.942201
AbstractPlus | Full Text: PDF(596 KB) IEEE CNF
Rights and Permissions

Google Scholar BETA

Web Images Video News Maps more »

folder categories 1996 "e mail " Search

Advanced Scholar Search
Scholar Preferences
Scholar Help

Scholar **All articles - Recent articles** Results **1 - 10** of about **13,500** for **folder categories 1996 "e m**

All Results
O White
K Nelson
W Cohen
M Daly
J Chan

Did you mean: folder categories 1996 "***email*** "

[PS] Learning rules that classify **e-mail** - all 11 versions »
WW Cohen - AAAI Spring Symposium on Machine Learning in Information ..., **1996** - parc.com
... modify it by replacing \95" with \96" and \1995" with **1996**". ... am reluctant to use most of the **categories** of the form \messages to be led in **folder** foo " as ...
Cited by 285 - Related Articles - View as HTML - Web Search

[PS] Automatic induction of rules for **e-mail** classification - all 5 versions »
E Crawford, J Kay, E McCreath - ADCS2001 Proceedings of the Sixth Australasian Document ... - cs.anu.edu.au
... We also believe that there are **categories** which can be ... examples may be available for some **folders** for some ... Learning in Information Access, pages 18{25, **1996**. ...
Cited by 31 - Related Articles - View as HTML - Web Search

MailCat: an intelligent assistant for organizing **e-mail** - all 8 versions »
RB Segal, JO Kephart - Proceedings of the third annual conference on Autonomous ..., 1999 - portal.acm.org
... MailCat: An Intelligent Assistant for Organizing **E-Mail** ... the similarity be- tween M and each **folder** 3. In ... a ranking of several possible **categories**, rather than ...
Cited by 125 - Related Articles - Web Search

The Enron Corpus: ANew Dataset for Email Classification Research - all 9 versions »
B Klimt, Y Yang - Machine Learning: ECML 2004: 15th European Conference on ..., 2004 - books.google.com
... 1], as macro-averages are dominated by small **categories**. ... they are all in the same **folder** (Deleted Items ... of the **1996** AAAI Spring Symposium in Information Access ...
Cited by 79 - Related Articles - Web Search

Method for accessing computer files and data, using linked **categories** assigned to each data file ... - all 3 versions »
J Lewak, S Grzechnik, J Matousek... - US Patent 5,544,360, **1996** - Google Patents
... 6, **1996** Sheet 4 of 4 5,544,360 ... own "hybrid **folders**" by simply describing, using **categories** the user ... of those files which are to belong to each "hybrid **folder**". ...
Cited by 74 - Related Articles - Web Search

... which utilizes a probabilistic classifier to detect" junk" **e-mail** by automatically updating a ... - all 3 versions »
E Horvitz, DE Heckerman, ST Dumais, M Sahami, JC ... - US Patent 6,161,130, 2000 - Google Patents
... MESSAGE j IN SPAM MAIL **FOLDER** 227 FOR ... on Machine Learning in Informa - tionAccess,
1996 (hereinafter the ... classified 20 into different **categories** is provided as ...
Cited by 81 - Related Articles - Web Search

Effectiveness of Parks in Protecting Tropical Biodiversity - all 12 versions »
AG Bruner, RE Gullison, RE Rice, GAB da Fonseca - Science, 2001 - sciencemag.org
... **E-mail**: a.bruner{at}conservation.org ... the entire range of outcomes (eg, four **categories** were used to ... J. Baillie, B. Groombridge, Eds., **1996** IUCN Red List of ...
Cited by 251 - Related Articles - Web Search

Plasma Insulin-Like Growth Factor-I and Prostate Cancer Risk: A Prospective Study - all 12 versions »
JM Chan, MJ Stampfer, E Giovannucci, PH Gann, J Ma ... - Science, 1998 - sciencemag.org
... **E-mail**: jmlchan{at}hsph.harvard.edu ... were considered "high grade/stage cancer." Also, we assigned that **category** to stage ... 88, 1118 (**1996**) [Abstract/Free Full Text ...
Cited by 1000 - Related Articles - Web Search

[PDF] Learning to Filter Spam **E-Mail**: A Comparison of a Naive Bayesian and a Memory-Based Approach - all 19 versions »
I Androutsopoulos, G Paliouras, V Karkaletsis, G ... - Arxiv preprint cs.CL/0009009, 2000 - arxiv.org
... recently started experimenting with suitably "encoded" personal **e-mail folders**. ... 8 Consult (Mitchell **1996**) for more elaborate ... 1 I = belongs to **category** c is ...
Cited by 142 - Related Articles - View as HTML - Web Search - Library Search

EMAIL OVERLOAD: EXPLORING PERSONAL INFORMATION MANAGEMENT OF EMAIL - all 20 versions »
S Whittaker - Culture of the Internet, 1997 - books.google.com
... inboxes, by leaving one semantic **category** exemplar in ... to mediate cognition (Norman, 1988; Walker, **1996**; Whittaker et aL ... We saw that email **folders** function as an ...
Cited by 363 - Related Articles - Web Search

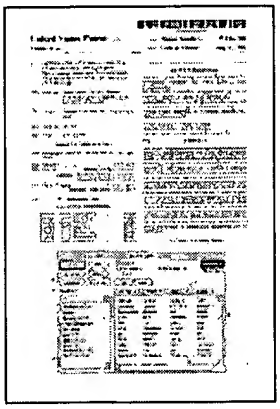
Did you mean to search for: folder categories 1996 "***email*** "

Gooooooooogle ▶
Result Page: **1** 2 3 4 5 6 7 8 9 10 **Next**

folder categories 1996 "e mail " Search

Google Home - About Google - About Google Scholar

Method for accessing computer files and data, using linked categories ... Jerzy Lewak et al



Abstract | Drawing | Description | Claims

Abstract
A computer filing system for accessing files and data according to user-designated criteria. The system allows the user to define a virtually unlimited number of hybrid folders by describing, using terms of their own selection, the file contents of those files which are to belong to particular hybrid folders. Such hybrid folders can be implemented on top of, and used in addition to, the normal hierarchical structured directory, or they may replace such normal structures entirely. The inventive computer file control system could therefore be used as the basis of a new computer operating system. In the process of search and retrieval, the invention ensures in two ways that the user defines a filter which will always find at least one file. The user is not required to type the key words to search but instead chooses the words from pick lists, making mistyping impossible. As the user builds the search filter definition, categories which would find no data are automatically excluded...

Patent number: 5544360
Filing date: Feb 3, 1995
Issue date: Aug 6, 1996
Inventors: Jerzy Lewak, Slawek Grzechnik, Jon Matousek

Claims

What is claimed is:

1. A method for accessing files in a data storage system of a computer system having means for reading and writing data from the data storage system, displaying information, and accepting user input, the method comprising the steps of:

(a) initially creating in the computer system a category description table containing a plurality of category descriptions, each category description comprising a descriptive name, the category descriptions having no predefined hierarchical relationship with such list or each other;
(b) thereafter creating in the computer system a file information directory comprising at least one entry corresponding to a file on the data storage system, each entry comprising at least a unique file identifier for the corresponding file, and a set of category descriptions selected from the category description table; and
(c) thereafter creating in the computer system a search filter comprising a set of category descriptions, wherein for each category description in the search filter there is guaranteed to be at least one entry in the file information directory having a set of category descriptions matching the set of category descriptions of the search filter.

2. A method for accessing files in accordance with claim 1, wherein each category description comprises a user defined category name and a unique category description identifier created by the computer system.

3. A method for accessing files in accordance with claim 2, wherein each category description further comprises a category type designation.

4. A method for accessing files in accordance with claim 2, wherein the step of creating a category description table comprises the steps of:

(1) accepting user input defining a new category description;
(2) displaying the new category description;
(3) creating a unique category description identifier associated with the new category description; and

Assignee: Paragon Concepts, Inc.
Primary Examiner: Jack M. Choules

U.S. Classification
395/600; 364/253; 364/253.3; 364/282.1; 364/282.3; 364/283.1; 364/283.2; 364/283.3; 364/286.4; 364/286.5

International Classification
G06F 1730

Search within this patent

Search

Citations

Patent Number	Title	Issue date
5047918	File management system	Sep 10, 1991
5115504	Information management system	May 19, 1992
5162992	Vector relational characteristical object	Nov 10, 1992
5201047	Attribute-based classification and retrieval system	Apr 6, 1993
5201048	High speed computer system for search and retrieval of data within text and record oriented files	Apr 6, 1993
5204947	Application independent (open) hypermedia enablement services	Apr 20, 1993

5206949	Database search and record retrieval system which continuously displays category names during scrolling and selection of individually displayed search terms	Apr 27, 1993
5276867	Digital data storage system with improved data migration	Jan 4, 1994

Referenced by

Patent Number	Title	Issue date
5715444	Method and system for executing a guided parametric search	Feb 3, 1998
5715449	Method for generating structured medical text through user selection of displayed text and rules	Feb 3, 1998
5724577	Method for operating a computer which searches a relational database organizer using a hierarchical database outline	Mar 3, 1998
5734901	Electronic mail information associated with native application data	Mar 31, 1998
5761680	Coherent film system access during defragmentation operations on a storage medium	Jun 2, 1998
5787446	Sub-volume with floating storage space	Jul 28, 1998
5794233	Browse by prompted keyword phrases	Aug 11, 1998
5797139	Method, memory and apparatus for	Aug 18, 1998

	designating a file's type by building unique icon borders	
5819269	Dynamic subgrouping in a news network	Oct 6, 1998
5826254	System for selectively browsing a large, distributed directory tree using authentication links	Oct 20, 1998
5832513	Detecting significant file system alterations during execution of a storage media software utility	Nov 3, 1998
5878408	Data management system and process	Mar 2, 1999
5890147	Scope testing of documents in a search engine using document to folder mapping	Mar 30, 1999
5905990	File system viewpath mechanism	May 18, 1999
5913215	Browse by prompted keyword phrases with an improved method for obtaining an initial document set	Jun 15, 1999
5924104	Method and apparatus for displaying intradocument links in a computer system	Jul 13, 1999
5948058	Method and apparatus for cataloging and displaying e-mail using a classification rule preparing means and providing cataloging a piece of e-mail into multiple categories or classification types based on e-mail object information	Sep 7, 1999
5970491	System and method of storage management for an electronic mail system	Oct 19, 1999
5974415	System and method for computer-aided	Oct 26, 1999

	heuristic adaptive attribute matching	
5983219	Method and system for executing a guided parametric search	Nov 9, 1999
6098071	Method and apparatus for structured document difference string extraction	Aug 1, 2000
6119127	Game software management system, linking game files	Sep 12, 2000
6148294	System and method for computer directory updating and presentation based on frequency of access	Nov 14, 2000
6185574	Multiple display file directory and file navigation system for a personal computer	Feb 6, 2001
6240414	Method of resolving data conflicts in a shared data environment	May 29, 2001
6275821	Method and system for executing a guided parametric search	Aug 14, 2001
6282548	Automatically generate and displaying metadata as supplemental information concurrently with the web page, there being no link between web page and metadata	Aug 28, 2001
6324538	Automated on-line information service and directory, particularly for the world wide web	Nov 27, 2001
6327588	Method and system for executing a guided parametric search	Dec 4, 2001
6389427	File system performance enhancement	May 14, 2002

6393435	Method and means for evaluating the performance of a database system referencing files external to the database system	May 21, 2002
6427123	Hierarchical indexing for accessing hierarchically organized information in a relational system	Jul 30, 2002
6457017	Computing system for information management	Sep 24, 2002
6463428	User interface providing automatic generation and ergonomic presentation of keyword search criteria	Oct 8, 2002
6484164	Data search user interface with ergonomic mechanism for user profile definition and manipulation	Nov 19, 2002
6496837	Multiple attribute file directory manipulation and navigation system	Dec 17, 2002
6499029	User interface providing automatic organization and filtering of search criteria	Dec 24, 2002
6505194	Search user interface with enhanced accessibility and ease-of-use features based on visual metaphors	Jan 7, 2003
6519612	Internet storage manipulation and navigation system	Feb 11, 2003
6526410	Method and apparatus for structured document difference string extraction	Feb 25, 2003
6549916	Event notification system tied to a file system	Apr 15, 2003

6571231	Maintenance of hierarchical index in relational system	May 27, 2003
6571235	System for providing an interface for accessing data in a discussion database	May 27, 2003
6633312	Method and apparatus for selecting network entities	Oct 14, 2003
6662177	Search user interface providing mechanism for manipulation of explicit and implicit criteria	Dec 9, 2003
6671692	System for facilitating the navigation of data	Dec 30, 2003
6671693	System for effectively collecting and disseminating data	Dec 30, 2003
6675161	Managing changes to a directory of electronic documents	Jan 6, 2004
6690992	Game software management system, linking game files	Feb 10, 2004
6694339	File management device and method thereof, and audio visual data recording/reproducing device and method thereof	Feb 17, 2004
6694308	System and method for user adaptive software interface	Feb 17, 2004
6725228	System for managing and organizing stored electronic messages	Apr 20, 2004
6751604	Method of displaying temporal and storage media relationships of file names protected on removable storage media	Jun 15, 2004
6778972	System and method for providing integrated management of electronic information	Aug 17, 2004
6826559	Hybrid category mapping for on-line	Nov 30, 2004

	query tool	
6826566	Identifier vocabulary data access method and system	Nov 30, 2004
6826574	Automatic profiler	Nov 30, 2004
6829747	Editing apparatus and editing method	Dec 7, 2004
6834285	Computer system for portable digital data capture and data distribution	Dec 21, 2004
6862592	Document processing in a cross-platform environment	Mar 1, 2005
6880008	SYSTEM AND METHOD FOR RETRIEVING A BACKUP FILE HAVING A FILE NAME EXACTLY CORRESPONDING TO THE DATE AND TIME OF A CLIENT REQUEST, IF NO EXACT MATCH FOR THE DATE AND TIME CORRESPONDING TO THE CLIENT REQUEST IS FOUND, TO SEARCH FOR THE BACKUP FILE HAVING THE FILE NAME WITH A DATE AND TIME THAT ARE CLOSEST TO BUT PRIOR TO THE SPECIFIED DATE AND TIME	Apr 12, 2005
6922708	File system that supports transactions	Jul 26, 2005
6941302	Managing changes to a directory of electronic documents	Sep 6, 2005
6947936	Method for a topic hierarchy classification system	Sep 20, 2005
7024416	Semi-automatic index term augmentation in document retrieval	Apr 4, 2006
7028034	Method and apparatus for providing a	Apr 11, 2006

	dynamically-updating pay-for-service web site	
7047242	Weighted term ranking for on-line query tool	May 16, 2006
7047257	Computer file management system	May 16, 2006
7051277	Automated assistant for organizing electronic documents	May 23, 2006
7058648	Hierarchy-based secured document repository	Jun 6, 2006
7065523	Scoping queries in a search engine	Jun 20, 2006
7127464	Method for updating personal financial information on a web site	Oct 24, 2006
7146362	Method and apparatus for using faceted metadata to navigate through information resources	Dec 5, 2006
7155668	Method and system for identifying relationships between text documents and structured variables pertaining to the text documents	Dec 26, 2006
7236972	Identifier vocabulary data access method and system	Jun 26, 2007
7240329	Policies on a per instance basis	Jul 3, 2007
7260619	Method and apparatus for supplying information, and storage medium on which an information supplying program is stored	Aug 21, 2007
7263522	System and method for user adaptive software interface	Aug 28, 2007
7266592	Method and apparatus for supplying information, and storage medium on which an information supplying	Sep 4, 2007

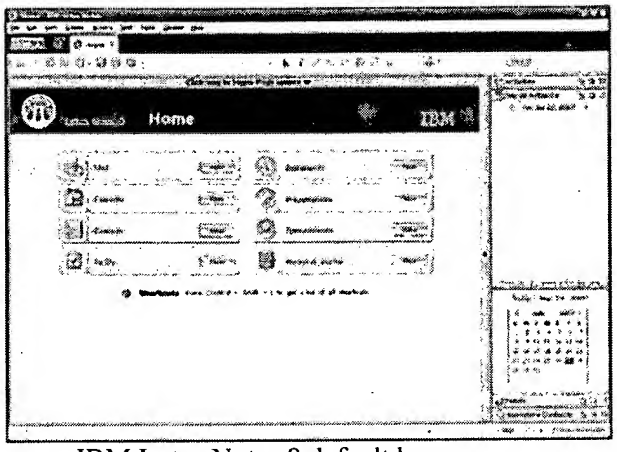
	program is stored	
7269591	Method and apparatus for providing a pay-for-service web site	Sep 11, 2007
7275063	Computer system for automatic organization, indexing and viewing of information from multiple sources	Sep 25, 2007

IBM Lotus Notes

From Wikipedia, the free encyclopedia
(Redirected from Lotus Notes)

Lotus Notes is a client-server, collaborative application owned by IBM Software Group. IBM defines the software as an "integrated desktop client option for accessing business e-mail, calendars and applications on [an] IBM Lotus Domino server."[1]

Lotus Notes



IBM Lotus Notes 8 default home screen

Developer	IBM
Stable release	8.0 (http://en.wikipedia.org/w/index.php?title=Template:Latest_stable_release/&action=edit) (August 17, 2007) [+/- (http://en.wikipedia.org/w/index.php?title=Template:Latest_stable_release/Lotus_Notes&action=edit&preload=Template:LSR/syntax)]
Preview release	8.0 Beta 3 (Public) (http://en.wikipedia.org/w/index.php?title=Template:Latest_preview_release/&action=edit) (March 15, 2007) [+/- (http://en.wikipedia.org/w/index.php?title=Template:Latest_preview_release/Lotus_Notes&action=edit&preload=Template:LPR/syntax)]
OS	Cross-platform
Genre	Collaborative software
License	Proprietary
Website	IBM Lotus Notes (http://www.ibm.com/software/notes)

Features

The Notes client is mainly used as an email client, but also acts as an instant messaging client (for Lotus Sametime), browser, notebook, and calendar/resource reservation client, as well as a platform for interacting with collaborative applications.

In the early days of the product, the most common applications were threaded discussions and simple contact management databases. Today Notes also provides blogs, wikis, RSS aggregators, CRM and Help Desk systems, and organizations can build a variety of custom applications for Notes using Domino Designer.

The Notes client can be used as an IMAP and POP e-mail client with non-domino mail servers. Recipient addresses can be retrieved from any LDAP server, including Active Directory. The client also does web browsing although most people configure it to launch their default browser instead.

Features include group calendaring and scheduling, SMTP-based e-mail (HTML based e-mail is available to Java developers), NNTP-based news support, and automatic HTML conversion of all documents by the Domino HTTP task.

Notes instant messaging (Sametime) allows you to see your coworkers online and have chat sessions with them. A chat session can be with one person or multiple people (an instant meeting). Beginning with Release 6.5 this functionality is built into the client and presence awareness has become more and more pervasive throughout the user experience.

Since version 7, Notes has provided a web services interface. Domino can be a web server for HTML files too; authentication of access to Domino databases or HTML files uses Domino's own user directory and external systems such as Microsoft's Active Directory.

A design client is available to allow rapid development of databases consisting of forms, which allow users to create documents; and views, which display selected document fields in columns.

In addition to being a "groupware" system (e-mail, calendaring, shared documents and discussions), Notes/Domino is also a platform for developing customized client-server and web applications. Its use of design constructs and code provide capabilities that facilitate the construction of "workflow" type applications (which may typically have complex approval processes and routing of data).

Since Release 5, Lotus server clustering has been capable of providing geographic redundancy for servers.

Data replication

A generalized replication facility was implemented in the first release of Notes. The generalized nature of this feature set it apart from predecessors like Usenet, and continues to differentiate Notes from many other systems that now offer some form of synchronization or replication. The facility in Notes and Domino is not limited to email, calendar, and contacts. It works for any data in any application that uses NSF files, which are the standard container for data in the Notes architecture, for its storage. No special programming, tagging, or other configuration is required to enable replication.

Domino servers and Notes clients identify NSF files by their Replica IDs and keep files with matching IDs synchronized by bidirectionally exchanging data, metadata, and application logic and design. Replication between two servers, or between a client and a server, can occur over a network or a point-to-point modem connection. Replication between servers may occur at intervals according to a defined schedule, in near real-time when triggered by data changes in Domino server clusters, or on an ad hoc basis when triggered by an administrator or programmatically.

Creation of a local replica of an NSF file on the hard disk of a Notes client enables the user of the client to take full advantage of Notes databases while working off-line—with the client synchronizing any changes when client and server next connect. Local replicas are also sometimes maintained for use while connected to the network in order to reduce network i/o. Replication between a Notes client and Domino server can run automatically according to a schedule, or manually in response to a user or programmatic request. Local replicas on early releases of the Notes client did not always maintain all security features programmed into the applications, but starting with Notes 6 enforcement of application security is automatic for all local replicas. Early releases also did not offer a way to encrypt NSF files, raising concerns that local replicas potentially exposed too much confidential data on laptop computers or insecure home office computers, but an optional encryption feature for NSF files was added in more recent releases, and as of Notes 6 it is the default setting for newly created local replicas.

Security

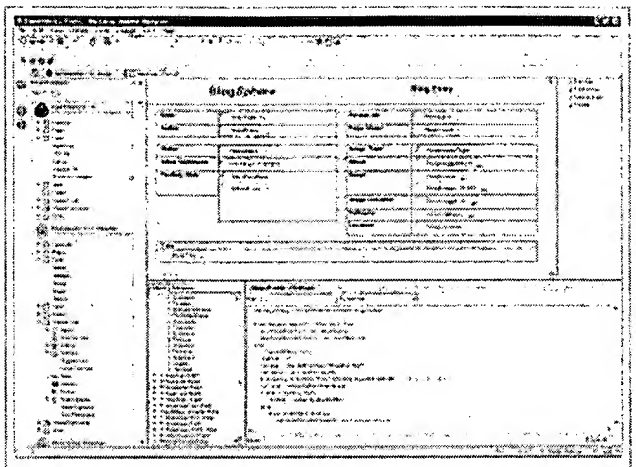
Security is built into the product. Notes was the first widely adopted software product to use public key cryptography for client-server and server-server authentication and for encryption of data, and it remains the product with the largest installed base of PKI users. Until US laws regulating encryption were changed in 2000, Lotus was prohibited from exporting versions of Notes that supported symmetric encryption keys that were longer than 40 bits. In 1997, Lotus negotiated an agreement with the NSA that allowed export of a version that supported stronger keys with 64 bits, but 24 of the bits were encrypted with a special key and included in the message to provide a "workload reduction factor" for the NSA. The effect of this was that users of Notes outside of the US had stronger protection against private sector industrial espionage, but no additional protection against spying by the US government.[2] This implementation was not a secret - in fact it was widely announced - but with some justification many people did consider it to be a backdoor. Some governments objected to being put at a disadvantage to the NSA, and as a result Lotus continued to support the 40 bit version for export to those countries. Under current US export laws, Lotus supports only one version of the Notes PKI with 128 bit symmetric keys, 1024 bit public keys, and no workload reduction factor. The Domino server's security tools also include S/MIME, SSL 3.0 support with industry standard key sizes for HTTP and other Internet protocols, X.509 client certificates, and an integrated certificate authority.

Lotus also employs a code-signature framework that controls the security context, runtime, and rights of custom code developed and introduced into the environment. With Release 5, Lotus introduced

Execution Control Lists at the Client level - starting with 6, ECL's can be managed centrally by the SA's through the implementation of Policies. Code signatures are widely regarded as the reason there has not been a virus capable of propagating natively through a Notes/Domino environment since release 4.5.

Programming

Notes/Domino is a cross-platform, secure, distributed document-oriented database and messaging framework and rapid application development environment that includes pre-built applications like email, calendar, etc. This sets it apart from its major commercial competitors, such as Microsoft Exchange or Novell GroupWise, which are generally purpose-built applications for mail and calendaring that offer APIs for extensibility.



Domino Designer 7 IDE.

Lotus Domino databases are built using the Domino Designer client, available only for Windows. A key feature of Notes is that many replicas of the same database can exist at the same time on different servers and clients, across dissimilar platforms, and the same storage architecture is used for both client and server replicas. Originally, replication in Notes happened at document (i.e. record) level. With release of Notes 4 in 1996, replication was changed so that it now occurs at field level.

A database is an NSF (Notes Storage Facility) file, containing basic units of storage known as a "note". Every note has a UniqueID and a NoteID. The UniqueID uniquely identifies the note across all replicas within a cluster of servers, a domain of servers, or even across domains belonging to many organizations that are all hosting replicas of the same database. The NoteID, on the other hand, is unique to the note only within the context of one given replica. Each note also stores its creation and modification dates, and one or more Items.

There are several classes of notes, including design notes and document notes. Design notes, which are created and modified with the Domino Designer client, represent programmable elements, such as the GUI layout of forms for displaying and editing data, or formulas and scripts for manipulating data. Document notes, which are created and modified with the Lotus Notes client, via a web browser, via mail routing and delivery, or via programmed code, represent user data.

Document notes can have parent-child relationships, but Notes should not be considered a hierarchical database in the classic sense of IMS. Notes databases are also not relational, although there is a SQL driver that can be used with Notes, and it does have some features that can be used to develop applications that mimic relational features. There is no support for atomic transactions in Notes, and its file locking is rudimentary at best. Notes is essentially a document-based, schemaless, loosely structured database with support for rich content and powerful indexing facilities. This structure closely mimics paper-based workflows that Lotus Notes is typically used to automate.

Items represent the content of a note. Every item has a name, a type, and may optionally have some flags set. A note can have more than one item with the same name. Types include Number, Number List,

Text, Text List, Date-Time, Date-Time List, and Rich Text. Flags are used for managing attributes associated with the item, such as read or write security. Items in design notes represent the programmed elements of a database. For example, the layout of an entry form is stored in the rich text Body item within a form design note. This means that the design of the database can replicate to users' desktops just like the data itself, making it extremely easy to deploy updated applications.

Items in document notes represent user-entered or computed data. An item named "Form" in a document note can be used to bind a document to a form design note, which directs the Lotus Notes client to merge the content of the document note items with the GUI information and code represented in the given form design note for display and editing purposes. The resulting loose binding of documents to design information is one of the cornerstones of the power of Lotus Notes. Traditional database developers used to working with rigidly enforced schemas, on the other hand, may consider the power of this feature to be a double-edged sword.

Notes applications development uses several programming languages. Formula and LotusScript are the two main ones. LotusScript is similar to, and may even be considered a specialized implementation of, Visual Basic, but with the addition of many powerful native classes that model the Notes environment, whereas Formula is unique to Notes but similar to Lotus 1-2-3 formula language.

Since Release 5, Java and JavaScript are also integrated into Lotus Notes. LotusScript is the primary tool in developing applications for the Notes client, as well as server-based processing. Java and JavaScript are the primary tools for developing applications for browser access, allowing browsers to emulate the functionality of the Notes client. The Notes client can now natively process Java and JavaScript code, although applications development usually requires at least some code specific to only Notes or only a browser. However, the Mac client does not support Java and the Windows client usually does not support the most recent version of Java.

As of version 6, Lotus established an XML programming interface in addition to the options already available. The Domino XML Language (DXL) provides XML representations of all data and design resources in the Notes model, allowing any XML processing tool to create and modify Notes/Domino data.

External to the Lotus Notes application, IBM provides toolkits in C, C++, and Java to connect to the Domino database and perform a wide variety of tasks. The C toolkit is the most mature and the C++ toolkit is an objectized version of the C toolkit, lacking many functions the C toolkit provides. The Java toolkit is the least mature of the three and can be used for basic application needs.**

Database

Notes includes a DBMS but Notes files are different from relational or object databases since they are document centric, allow multiple values in items (fields), don't require a schema, come with built-in document-level access control and store RichText data. There are some Object-Relational features being developed and Domino 7 supported a restricted release add-on allowing the use of IBM's DB2 database as an alternative store for Notes databases. With Domino 8 this is available without a special request. You can map a Notes database to a relational database using tools like DECS, [LEI], JDBCSql for Domino or NotesSQL (http://www-142.ibm.com/software/sw-lotus/products/product4.nsf/wdocs/notessqlhome).

It could be argued that Notes is a multi-value database system like PICK, or that it's an object system like Zope, but it is in fact unique. Whereas the temptation for relational database programmers is to normalize databases, Notes databases must be *de-*normalized. RDBMS developers often find it difficult to conceptualize the difference. It may be useful to think of a Notes document (a 'note') as analogous to an XML document natively stored in a database (although with limitations on the data types and structures available).

The benefits of this data structure are:

1. No need to define size of fields, or datatype — although you can if you want to;
2. Attributes (= Notes fields) that are null take up no space in a database;
3. Built-in full text searching.

Use as an email client

Lotus Notes is commonly deployed as an end-user email client in larger organizations, accounting for more than 120 million total users according to IBM's latest figures.[3]

When an organization employs a Lotus Domino server, it usually also deploys Lotus Notes for its users to read mail and use databases. However, the Domino server also supports POP3 and IMAP mail clients, and through an extension product (Domino Access for Microsoft Outlook) supports native access for Microsoft Outlook clients. Lotus also provides Domino Web Access, to allow the use of email and calendaring features through Internet Explorer and Firefox web browsers on Windows, Mac and Linux. There are several spam filtering programs available, and a rules engine allowing user-defined mail processing to be performed by the server.

How Notes differs from other email clients

Lotus Notes is a unique environment. It was designed to be a collaborative application platform where email was just one of numerous applications that ran in the Notes client software. Lotus lore has it that the first mail inbox application written by Lotus was a proof-of-concept for a sales presentation. The Notes client was also designed to run on multiple platforms including Windows, OS/2, Mac, SCO Open Desktop UNIX, and Linux. These two factors have resulted in the user interface containing some differences from applications that only run on Windows. Furthermore these differences have often remained in the product to retain backward compatibility with earlier releases, instead of conforming to Windows UI standards. The following are some of these differences.

- Properties dialog boxes for formatting text, hyperlinks and other rich-text information can remain open after changes are made to the selected text. This provides great flexibility to select new text and apply other formatting without closing the dialog box, selecting new text and opening a new format dialog box. Almost all other Windows applications require the user to close the dialog box, select new text, then open a new dialog box for formatting/changes.
- Properties dialog boxes also automatically recognize the type of text selected and display the appropriate selections, for instance, a hyperlink properties box when appropriate.
- Tables can be formatted as tabbed interfaces as part of form design (for applications) or by users within mail messages(or in rich-text fields in applications). This provides users the ability to provide tab-style organization to documents, similar to popular tab navigation in most web

portals, etc.
- Links to Notes applications, views or documents can be easily inserted into Notes documents.
- Full-text searching is nearly instantaneous compared to other email applications, including Outlook.

- Deleting a document (or email) will delete it from *every* folder in which it appears, since the folders simply contain links to the same back-end document. Some other email clients only delete the email from the current folder; if the email appears in other folders it is left alone, requiring the user to hunt through multiple folders in order to completely delete a message. In Notes, clicking on "Remove from Folder" will remove the document only from that folder leaving all other instances intact.

- The *All Documents* and *Sent* folders exhibit some different behaviors than other folders. Namely, you cannot drag email out of them and thereby remove the email from those folders; the email can only be "copied" from them. This is because these two folders are not, in fact, folders at all: they are views. Their membership indexes are maintained according to programmed criteria rather than user interaction (as with a folder). This technical difference is not apparent to some users, but it does make sense. *All Documents* contain all of the documents in your mail database, no matter which folder it is in. The only way to remove something from *All Documents* is to delete it outright. Also, an email message cannot be removed from *Sent*, since that would imply that a message that used to be sent was no longer ever sent, which does not make sense.

Lotus Notes 7 and older versions had more differences:

- Users select a "New Memo" to send an email, rather than "New Mail" or "New Message." (In Notes 8, the command is called "New Message.")

- To select multiple documents in a Notes view, you drag your mouse next to the documents that you want to select, rather than using ⇧ Shift +single click. (Notes 8 uses standard conventions.)

- The searching function is a "phrase search", rather than the more common "or search", and Notes requires users to spell out boolean conditions in search string. As a result, users must search for 'delete AND folder' in order to find help text that contains the phrase 'delete a folder'. Searching for 'delete folder' does not yield the desired result.

- The built-in full text search engine will only find email in the currently selected folder or view; if you click search while you are in your inbox, then that's the only place that the search will look. To the user, it can appear that Notes cannot find the email, when in fact, the user is simply "not looking in the correct place". (If you want to search the entire mailbox, the correct place to initiate the search is the Notes mailbox's *All Documents* view.)

- Notes up to version 7 uses F9 as its refresh key and F5 to lock the display. Some Microsoft applications (e.g., Outlook 2002, Explorer, Internet Explorer) use F5 as a refresh key, others (e.g. Outlook 2003, Word, Excel) use F9. F9 's use as the refresh key in PC applications pre-dates Microsoft's choice of F5, back to the early 1980s, when Lotus 1-2-3 was the most popular PC application. In Notes 8, both F9 and F5 refresh the screen, and Ctrl + F5 is used to lock Notes.

- The delete key does not immediately delete a document, but rather marks it for deletion later
- Pressing the delete key on a deleted item will effectively undelete it

Like all popular commercial software packages, Lotus Notes has its detractors (http://lotusnotessucks.4t.com/) as well as supporters (http://www.openntf.org/). Critics assert that there are dedicated email clients that are simpler, more intuitive and have a lower purchase price. Proponents argue that richer capabilities and advanced programmability are available, and that purchase price is a small fraction of total cost of ownership. Many of the differences mentioned above are seen by some as weaknesses in the product, especially when the user interface is compared to Windows-only applications.

Later releases of the product have made great headway in addressing end-user complaints.

Notes 8.0 (released in 2007) was the first version that employed a full user-experience team Mary Beth Raven Blog (http://www-03.ibm.com/developerworks/blogs/page/marybeth), resulting in a greatly improved Notes client experience. Additionally, Notes 8.0 now runs in the open source Eclipse Software Framework, opening up nearly unlimited application development opportunities through the use of Eclipse plug-ins. The improved user experience builds on Notes 6.5 (released in 2003), which upgraded the e-mail client, previously regarded as the product's Achilles heel. Features added at that time included:

- drag and drop of folders
- replication of unread marks between servers
- follow-up flags
- reply and forward indicators on emails
- ability to edit an attachment and save the changes back to an e-mail

Followers of the corporate email strategies who have knowledge of Microsoft teams previously dedicated to converting companies from Lotus Notes to Outlook report that these teams have been disbanded. Gartner and other industry analysts now conclude that full scale conversions from Lotus Notes/Domino to Outlook/Exchange architecture has no benefit to organizations.

Criticisms

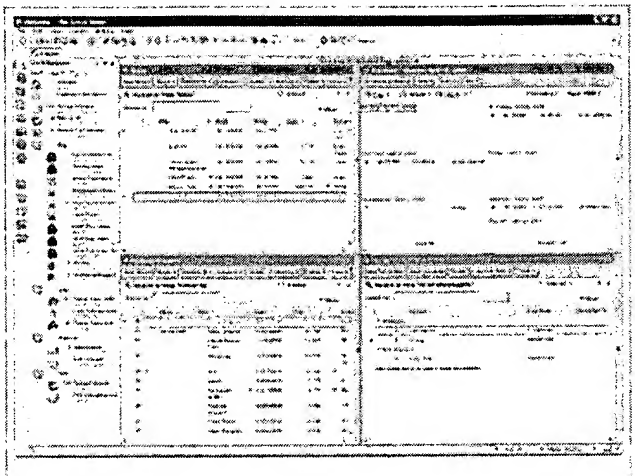
Criticisms of the product include:

- Many end users, particularly those mainly using it for emailing, have complained that aspects of the graphical user interface are unintuitive.[4] [5] Some of these criticisms have been addressed in recent versions.
- End users also complain about the excessive number of mouse clicks needed to perform basic functions, and the poor design of keyboard alternatives
- Prior to Release 8.0 the out of office feature was designed to limit traffic by sending the out of office messages at 2am (by default) instead of immediately after messages are received, giving the impression that it did not work. Recent releases have increased the frequency of out of office emails, which are now sent every few hours. In 8 this feature can be configured by administrators to run in real time enabling users to set their out of office and have it take effect immediately.
- Setting up archiving for the first time was complex, and often did not create an archive file straight away (the file is created when the first email is archived). This may have led some users to believe it did not work.
- Some users get confused between the "Starts with" search and the full text search features offered in the Notes UI, the former of which will only search on data that is visible in the currently sorted

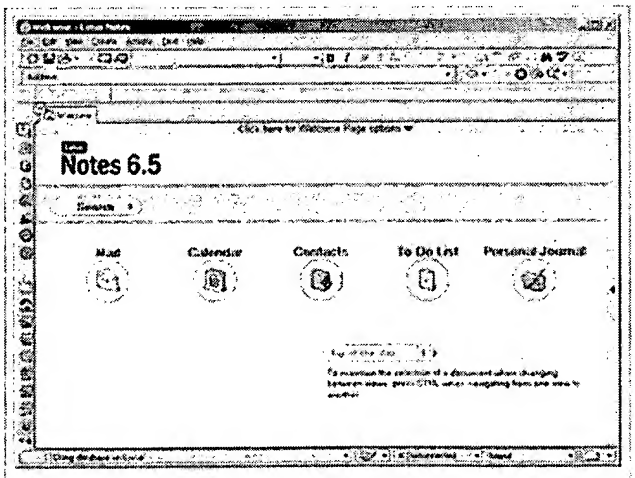
column of the visible folder. The product's defenders point out that the optional full text search feature has provided superior searching in Notes since the early 1990s, whereas rival Microsoft Outlook only provided that feature starting with the 2007 version, relying on third-party external plug-ins to provide similar functionality before that.
- It is difficult to read your Notes email from a standalone NSF file with any other software.
- Double clicking an attachment in Lotus Notes does not immediately open the attachment. A dialog box opens up with options to open the attachment or to view the size.
- It is hard to get the regular email address of someone that has sent an email to you if this person is from your Organization (in the Notes directory) because Notes does not display email addresses but specific Notes addresses like "Mike Smith/San Jose/ABC Company"

History

Lotus Notes has a history spanning more than 20 years.[6] Its chief inspiration was *PLATO Notes*, created by David Woolley at the University of Illinois in 1973. In today's terminology, PLATO Notes was a message board, and it was the basis for an online community which thrived for more than 20 years on the PLATO system. Ray Ozzie, who in 2006 succeeded Bill Gates as Chief Software Architect at Microsoft, worked with PLATO while attending the University of Illinois in the 1970s. When PC network technology began to emerge, Ozzie made a deal with Mitch Kapor, the founder of Lotus Development Corporation, that resulted in the formation of Iris Associates in 1984 to develop products that would combine the capabilities of PCs with the collaborative tools pioneered in PLATO. The agreement put control of product development under Ozzie and Iris, and sales and marketing under Lotus. In 1994, after the release and marketplace success of Notes R3, Lotus purchased Iris. In 1995 IBM purchased Lotus.



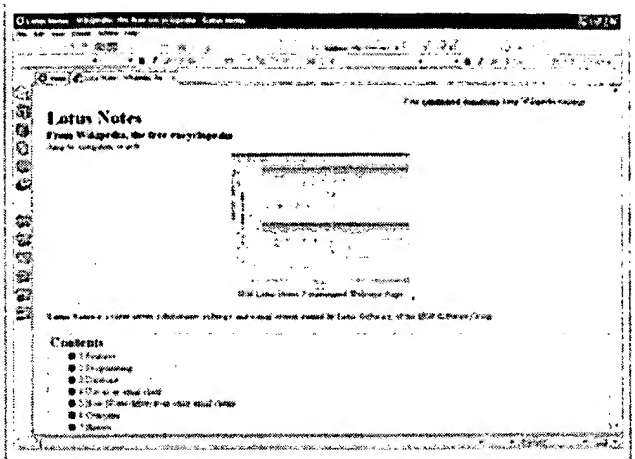
IBM Lotus Notes 7 customized Welcome Page



IBM Lotus Notes 6.5 default Welcome Page

When Lotus Notes was initially released, the name "Notes" referred to both the client and server components. In 1996, Lotus released an HTTP server add-on for the Notes 4 server called "Domino". This add-on allowed Notes documents to be rendered as web pages in real time. Later that year, the Domino web server was integrated into release 4.5 of the core Notes server and the entire server program was re-branded, taking on the name "Domino". Only the client program officially retained the "Lotus Notes" name, however end users are generally unaware of this differentiation, so even though more than ten years has passed since the re-branding, references to the "Lotus Notes Server" are still

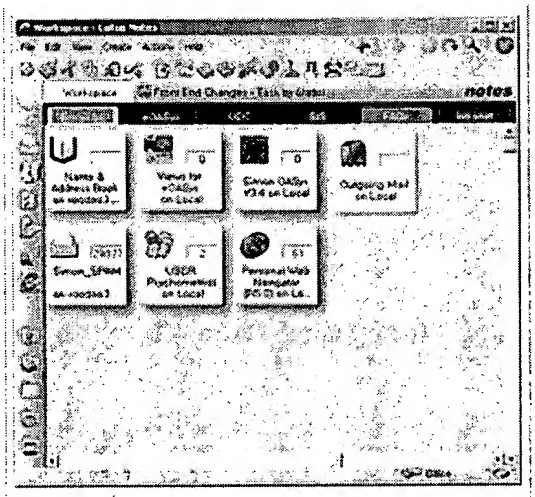
fairly common.



Lotus Notes Wikipedia article in Lotus Notes 6.5

Versions

- Release 1 - 1989 - The Notes client required DOS 3.1 or OS/2. The Notes server required either DOS 3.1, 4.0, or OS/2.
- Release 1.1 - 1990
- Release 2 - 1991
- Release 3 - May 1993
- Release 4 - January 1996
- Release 4.5 - December 1996. Server renamed as "Domino", added native HTTP server, POP3 (POP) server, added Calendaring & Scheduling. Also included SMTP MTA "in the box", but not installed by default.
- Release 4.6: added IMAP support. OS/2 and Unix client support dropped. No Mac client for this particular release.
- Release 5 - 1999: Moved SMTP functionality from a separate MTA task to become a native ability of the mail routing task, improving performance and fidelity of internet email. Major improvements to HTTP server. Notes client had a major interface overhaul.
- Release 5.0.8 - Added a new webmail interface, called iNotes (later changed to Domino Web Access in Release 6).
- Release 6.0 - September 2002. Added Domino Web Access (formerly iNotes Web Access) support. Dropped OS/2 server support.



Lotus Notes 5 client workspace

- Release 6.5 - September 2003. Added Lotus SameTime Instant Messaging integration to the Notes client (Windows only).
- Release 6.5.1 - January 2004
- Release 6.5.2 - June 2004
- Release 6.5.3 - November 2004
- Release 6.5.4 - April 2005
- Release 6.5.5 - December 2005
- Release 6.5.6 - March 2007
- Release 7.0 - August 2005. Added DB2 support as database storage
- Release 7.0.1 - July 2006. Added native Linux client, with initial release (http://www.ibm.com/software/swnews/swnews.nsf/n/nhan6rfrrb?OpenDocument&Site=software) certified for RHEL.
- Release 7.0.2 - September 2006. Added blog template, rss feed support, iCal support, SAP integration and "Nomad" which allows you to take your Notes client with you on a USB device.
- Release 7.0.3 - October 2007.
- Release 8.0 - August 2007

Current server versions available:

- Release 7.0.3 for All Platforms (Windows, Linux (Red Hat & SuSE x86, and zSeries), i5OS,

z/OS, Solaris 9 & 10)
- Release 8.0 for Windows, Linux, Solaris, AIX

Current client versions available:

- Release 7.0.3 for Windows
- Release 7.0.3 for Mac OS X
- Release 7.0.3 for Linux/x86 (Red Hat & SuSE initially). Various versions of the client have been run under Wine on Linux, but with varying degrees of success and no official support. The Notes 7 client and Domino Designer 7 are known to install and runs well under version 0.9.19.[7]. Domino servers can also translate most databases into HTML for browser based users.
- Release 8.0 for Linux and Windows XP/Vista English.

- Lotus notes code donation to OpenOffice.org - [1] (http://arstechnica.com/journals/linux.ars/2007/09/10/ibm-to-contribute-lotus-notes-code-to-openoffice-org) on September 12, 2007.

Future

Since the IBM acquisition of Lotus, some industry analysts and mainstream business press writers, along with IBM competitors, have made predictions of the impending demise of Lotus Notes. One noted example of this was an article published in *Forbes* magazine entitled "The decline and fall of Lotus", published in April 1998. Since that time, IBM claims that the installed base of Lotus Notes has nearly tripled from an estimated 42 million seats in September 1998 to more than 125 million in 2006.[3]

Speculation about the decline of Notes was fueled by lingering market confusion emanating from IBM placing marketing emphasis on Websphere and IBM Workplace in 2003 and 2004. IBM Workplace, however, has been discontinued[8], thus this source of confusion about the future of Notes and Domino has been rendered moot. While the future of any product in the technology sector cannot be predicted, IBM has made announcements that indicate that it continues to invest heavily in research and development on the Lotus Notes product line.

Notes 8, which was previously code-named "Hannover" (after the location of the 22nd Deutsche Notes User Group meeting, where it was first shown to the public) incorporates Notes into a larger Eclipse framework and includes support for productivity editors based on the OpenDocument format.[9] (These editors have also been released in a standalone package called Lotus Symphony.) In addition, IBM executive Ken Bisconti has made public comments on several occasions asserting that there will be releases 8, 9 and 10 of Notes and Domino.[10]

In 2005, some analysts concluded that Lotus is losing market share to Microsoft Exchange.[11] There is no general agreement, however, about methods of accurately calculating share in the messaging and collaboration market.[12] Figures based on seat count may be skewed by the presence of unused seats that are counted as a result of "bundled CALs", and figures based on customer count may be skewed by difference in typical customer organization sizes. IBM has asserted that growth shown in the revenue figures for the Lotus brand, as published in their audited annual financial report, show the continuing strength of the Lotus Notes product in the market. According to these figures, the Notes and Domino product line has sustained double-digit growth since late 2004 and continuing through 2006, including 30% year-to-year growth in Q4 of 2006.

In 2007, it was announced that IBM would contribute some of the code it had developed for the integration of the OpenOffice.org suite into Lotus Notes 8 back to the project[13]. IBM also packaged its version of Open Office for free distribution as IBM Lotus Symphony, released in beta mode on September 17[14].

See also

- List of IBM products
- Comparison of e-mail clients
- IBM Lotus Domino

References

1. ^ "IBM's home page for Lotus Notes" (http://www-142.ibm.com/software/sw-lotus/notes) Definition of Lotus Notes
2. ^ "The Swedes discover Lotus Notes has key escrow!" (http://catless.ncl.ac.uk/Risks/19.52.html#subj1) *The Risks Digest*, Volume 19, Issue 52, 1997-12-24
3. ^ *a b* Stan Beer (2006-07-10). Lotus Notes for Linux arrives (http://www.itwire.com.au/content/view/4910/53/). iTWire. Retrieved on 2007-05-15.
4. ^ Lotus Notes Sucks (http://lotusnotessucks.4t.com/) Lotus Notes Sucks website
5. ^ Survival of the unfittest (http://technology.guardian.co.uk/weekly/story/0,,1705106,00.html) *Guardian Unlimited*, February 9, 2006
6. ^ Official history of Lotus Notes (http://www.ibm.com/developerworks/lotus/library/ls-NDHistory) IBM DeveloperWorks Web Site
7. ^ Support info for running Notes 7 on Wine (http://wiki.winehq.org/LotusNotes) The Official Wine Wiki
8. ^ "Software withdrawal and service discontinuance: IBM Workplace Messaging" (http://www-306.ibm.com/common/ssi/fcgi-bin/ssialias?infotype=an&subtype=ca&htmlfid=897/ENUS906-251&appname=isource) IBM Press Release December 12, 2006
9. ^ "IBM backs OpenDocument in Lotus Notes" (http://news.com.com/IBM+backs+OpenDocument+in+Lotus+Notes/2100-1012_3-6072931.html) CNET News.com, Article published in May, 2006
10. ^ "Lotus set to uphold the future of Notes" (http://www.computerworld.co.nz/news.nsf/news/1CEC245081A1BF26CC2570FF00148759) Article published in January, 2006
11. ^ "IBM In Denial Over Lotus Notes" (http://www.forbes.com/technology/2005/04/06/cz_dl_0406notes.html) Article published in April 2005
12. ^ "Response to Daniel Lyons: "IBM In Denial Over Lotus Notes"" (http://www.shared-spaces.com/blog/2005/04/response_to_dan.html) Blogpost by Michael Sampson, Research Director at Shared Spaces
13. ^ "IBM Joins OpenOffice.org Community" (http://symphony.lotus.com/software/lotus/symphony/buzzentry.jspa?selectedCategoryID=5&threadID=2191&tstart=0)
14. ^ "IBM Releases Office Desktop Software at No Charge to Foster Collaboration and Innovation" (http://symphony.lotus.com/software/lotus/symphony/buzzentry.jspa?selectedCategoryID=5&threadID=2581&tstart=0)

External links

- Lotus Notes Homepage (http://www.lotus.com/notes)
- Global Lotus User Group Web Site (http://www.lotususergroup.org/)
- Lotus Domino Product Documentation

(http://www.ibm.com/developerworks/lotus/documentation/domino/)
- Lotus Notes Product Documentation (http://www.ibm.com/developerworks/lotus/documentation/notes/)
- Lotus Developer Domain: Products, E-Zine, Downloads, Discussion Forums, Documentation (http://www-10.lotus.com/ldd)
- The History of Notes and Domino (http://www.ibm.com/developerworks/lotus/library/ls-NDHistory)
- *IBM Redbooks - Lotus* (http://www.redbooks.ibm.com/portals/Lotus) Books on Lotus Notes and Domino, written by IBM staff, IBM business partners and customers
- OpenNTF.org: Open Source Lotus Notes/Domino applications (http://www.openntf.org/)
- Plato: The Emergence of Online Community (http://www.thinkofit.com/plato/dwplato.htm) David R. Woolley's description of PLATO Notes.
- Lotus Notes Hints, Tips, and Tricks (http://www.alanlepofsky.net/alepofsky/alanblog.nsf/archive?openview&title=Notes&type=cat&cat=Notes&sort=I)

Retrieved from "http://en.wikipedia.org/wiki/IBM_Lotus_Notes"

Categories: All articles with unsourced statements | Articles with unsourced statements since May 2007 | Upcoming software | Groupware | Lotus software | Windows e-mail clients | Mac OS e-mail clients | OS/2 software | Usenet clients | Database management systems

@PJL SET JOB

00887597 95-36989

Scalable Internet resource discovery: Research problems and approaches

Bowman, C Mic; Danzig, Peter B; Manber, Udi; Schwartz, Michael F

Communications of the ACM v37n8 pp: 98-107+

Aug 1994

ISSN: 0001-0782 **Journal Code:** ACM

Document Type: Journal article **Language:** English **Length:** 11 Pages

Special Feature: Charts Graphs References

Word Count: 7931

Abstract:

The Internet was conceived primarily as a means of remote login and experimentation with data communication protocols. However, the predominant usage quickly became e-mail in support of collaboration. This trend continues into the present incarnation of the Internet, but with increasingly diverse support for collaborative data-sharing. E-mail has been supplemented by a variety of wide-area filings, information retrieval, publishing, and library access systems. At present, the Internet provides access to hundreds of gigabytes each of software, documents, sounds, images, and other file system data; library catalog and user directory data; and many other types of information. In the past few years, a number of resource discovery tools have been created and have gained wide popular acceptance on the Internet. The impact of scale on resource discovery tools is examined. The examination focuses on 3 scalability dimensions: 1. the burgeoning diversity of information systems, 2. the growing user base, and 3. the increasing volume of data available to users.

Text:

In its roots as the ARPANET, the Internet was conceived primarily as a means of remote login and experimentation with data communication protocols. However, the predominant usage quickly became email in support of collaboration. This trend continues into the present incarnation of the Internet, but with increasingly diverse support for collaborative data-sharing. Email has been supplemented by a variety of wide-area filing, information retrieval, publishing, and library access systems. At present, the Internet provides access to hundreds of gigabytes each of software, documents, sounds, images, and other file system data; library catalog and user directory data; weather, geography, telemetry, and other physical science data; and many other types of information.

To make effective use of this wealth of information, users need ways to locate information of interest. In the past few years, a number of such resource discovery tools have been created and have gained wide popular acceptance in the Internet [1, 9, 13, 17, 19, 22].(1) Our goal here is to examine the impact of scale on resource discovery tools, and place these problems into a coherent framework. We focus on three scalability dimensions: the burgeoning diversity of information systems, the growing user base, and the increasing volume of data available to users.

Table 1 summarizes these dimensions, suggests a set of corresponding conceptual layers, and indicates problems being explored by the authors, who comprise the Internet Research Task Force (IRTF) Research Group on Resource Discovery and Directory Service.(Table 1 omitted) Users perceive the available information at the information interface layer. This layer must support scalable means of organizing, browsing, and searching. The information dispersion layer is responsible for replicating, distributing, and caching information. This layer must support access information by a large, widely distributed user populace. The information-gathering layer is responsible for collecting and correlating the information from many incomplete, inconsistent, and heterogeneous repositories.

INFORMATION SYSTEM DIVERSITY

An important goal for resource discovery systems is to provide a consistent, organized view of information. Since information about a resource exists in many repositories -- within the object itself and within other information systems -- resource discovery systems need to identify a resource, collect information about it from several sources, and convert the representation to a format that can be indexed for efficient searching. As an example, consider the problem of constructing a user directory. In a typical environment, several existing systems contain information about users. The Sun NIS [23] database usually has information about a user's account name, personal name, address, group memberships, password, and home directory. The ruserd [23] server has information about a user's workstation and its idle time. Additionally, users often place information in a ".plan" file that might list the user's travel schedule, home address, office hours, and research interests.

As this example illustrates, information in existing Internet repositories has the following characteristics:

* It is heterogeneous: Each repository maintains the information it needs about resources. For example, the primary purpose of the NIS database is to maintain information about user accounts, while a user's ".plan" file often contains more personal information. Also, the two repositories represent the information differently: records in an NIS database have a fixed format, but a ".plan" file contains unstructured text.

* It is inconsistent: Most information contained in Internet repositories is dynamic. Certain properties change frequently, such as which workstations a person is using. Other properties change more slowly, such as a user's email address. Because information is maintained by several repositories that perform updates at different times using different algorithms, often the information in the various repositories will

conflict. For example, information about account name, address, and phone number may be maintained by both the NIS database and a user directory server. When a user's address or phone number changes, the directory service will probably be updated first. However, if the account changes, the NIS database will usually be the first to reflect the change.

* It is incomplete: Additional attributes of a resource can often be obtained by combining information from several repositories. For example, a bibliographic database does not contain explicit information about a person's research interests. However keywords might be extracted from the persons' research papers to infer research interests.

There are two common approaches to these information-gathering layer problems. The first approach -- data mapping -- generates an aggregate repository from multiple information sources. The second approach -- operation mapping -- constructs a "gateway" between existing systems; it maps the functionality of one system into another without actually copying the data.

DATA MAPPING

The first approach for accommodating diversity is to collect data from a set of underlying repositories and combine it into a homogeneous whole. Doing so involves two parts: mapping algorithms for collecting information, and agreement algorithms for correlating information [4].

A mapping algorithm is implemented as a function that collects information from a repository and reformats it. There may be several implementations of mapping algorithms, each customized for an existing repository. The most common mapping algorithms are implemented as clients of an existing service. For example, Netfind extracts user information from several common Internet services [22], including the Domain Naming System (DNS), the finger service, and the Simple Mail Transfer Protocol.

The agreement algorithm defines a method for handling conflicts between different repositories. For example, Figure 1 illustrates data for the Enterprise [2] user directory system, which is built on top of the Universe name service [4].(Figure 1 omitted) This figure shows three mapping algorithms that gather information from the NIS database, the ruserd server, and the user's email. Several attributes can be generated by more than one mapping algorithm. For example, address information exists potentially in both the NIS database and the information supplied by the user. The agreement algorithm considers information gathered directly from the user as the most reliable. Depending on the attribute, the agreement algorithm may permit some properties to have several values, such as the two address attributes that describe the user in Figure 1.
Data mining represents a special form of agreement algorithm, which works by cross-correlating information available from multiple repositories. This can have two benefits. First, it can flag inconsistencies. For example, Enterprise could inform users if it detected conflicts between the email addresses listed in different repositories. Second, data mining can deduce implicit information by cross-correlating existing information. For example, Netfind collects and cross-correlates data continuously from a number of sources, to form a far-reaching database of Internet sites. One

source might discover a new host called "astro.columbia.ed;" from DNS traversals, and cross-correlate this information with the existing site database record for columbia.edu ("columbia university, new york, new york"), its understanding of the nesting relationships of the Domain name space, and a database of departmental abbreviations, to derive a new record for the Astronomy Department at Columbia University.

Mapping and agreement algorithms operate best generally when they exploit the semantics of specific resource discovery applications. In the Netfind example described this was possible because the data were gathered from particular services, each with semantics that Netfind understands.

More generally, data gathering depends on some data type structure to help select semantically specific gathering methods. We are exploring a variety of data-typing approaches in the context of gathering file system data. We now consider briefly the problems that arise in typing and gathering this data.

To gather information effectively from file system data, it is helpful to extract information in different ways depending on file type. For example, while it is possible to index every token in program source code, the index will be smaller and more useful if it distinguishes the variables and procedures defined in the file. In contrast, applying this data-gathering algorithm to a program's associated textual documentation will not yield the most useful information, since it has a different structure. By typing data, information can be extracted in whichever way is most appropriate for each type of file.

File data can be typed explicitly or implicitly. Explicitly typing files has the advantage that it simplifies data gathering. An explicitly typed file conforms to a well-known structure, which supports a set of conventions for what data should be extracted. Explicit typing is most naturally performed by the user when a file is exported into the resource discovery system. We are exploring this approach in the Nebula file system [3] and Indie discovery tool [5].

Many files exist without explicit type information, as in most current anonymous FTP files.(2) An implicit typing mechanism can help in this case. For example, Essence [12] uses a variety of heuristics to recognize files as fitting into certain common classes, such as word processor source text or binary executables. These heuristics include common naming conventions or identifying features in the data. The MIT semantic File System uses similar techniques [11].

Given a typed file, the next step is to extract indexing information. This can be accomplished most easily through automatic content extraction, using a grammar that describes how to extract information from each type of file. For example, given a TeX word processing document, the grammar could describe where to extract author, title, and abstract information. For cases in which more complex data extraction methods are needed, one can provide an "escape" mechanism that allows arbitrary data extraction programs to be run.

Automatic extraction methods have the advantage that they can provide an

immediate base of usable information, but in general will generate some inappropriate keywords and miss generating other, desirable keywords. For this reason, it is prudent to augment these methods with means by which people can override the automatically extracted data.

In many cases file-indexing information can be extracted from each file in isolation. In some cases, however, it is useful to apply extraction procedures based on the relationships among files. For example, binary executable files can be indexed sometimes by gathering keywords from their corresponding documentation. One can use heuristics to exploit such implicit interfile relationships for common cases, augmented by means of specifying explicit relationships. For example, we are exploring an approach that allows users to create files in the file system tree that specify relationships among groups of files.

One final observation about data type structure is that the index should preserve type information to help identify context during searches. For example, keywords extracted from document titles can be tagged in the index so that a query will be able to specify that only data extracted from document titles should match the query. This stands in contrast to the approach (used by WAIS [13], for example) of allowing a free association between query keywords and extracted data. We discuss indexing schemes further later.

OPERATION MAPPING

A gateway between two resource discovery systems translates operations from one system into operations in another system. Ideally, the systems interoperate seamlessly, without the need for users to learn the details of each system. Sometimes, however, users must learn how to use each system separately.

Building seamless gateways can be hindered if one system lacks operations needed by another system's user interface [22]. For example, it would be difficult to provide a seamless gateway from a system (like WAIS) that provides a search interface to users, to a system (like Prospero [19]) that supports browsing only. Even if two systems support similar operations, building seamless gateways may be hindered by another problem: providing appropriate mappings between operations in the two systems. To illustrate the problem, consider the current interim gateway from Gopher [17] to Netfind, illustrated in Figure 2.(3)(Figure 2 omitted) Because the gateway simply opens a telnet window to a Unix program that provides the Netfind service, users perceive the boundaries between the two systems.

In contrast, we have built a system called Dynamic WAIS [12], which extends the WAIS paradigm to support information from remote search systems (as opposed to the usual collection of static documents). The prototype supports gateways to Archie and to Netfind, using the Z39.50 information

retrieval protocol to integrate the information spaces seamlessly. The Dynamic WAIS interface to Netfind is shown in Figure 3.(Figure 3 omitted)

The key behind the Dynamic WAIS gateways is the conceptual work of constructing the mappings between the WAIS search-and-retrieve operations

and the underlying Archie and Netfind operations. In the case of Netfind, for example, when the Dynamic WAIS user requests a search using the "dynamicnetfind.src" WAIS database, the search is translated to a lookup in the Netfind site database to determine potential domains to search. The Netfind domain selection request is then mapped into a WAIS database selection request (the highlighted selection in the XWAIS Question window). Once the user selects one of the domains to search, the WAIS retrieval phase is mapped into an actual domain search in Netfind (the uppermost window).

We are developing these techniques further, to support gateways to complex forms of data, such as scientific databases.

USER BASE SCALE

New constituencies of users will make the Internet grow significantly beyond its present 2 million nodes. This growth will overburden the network's resource discovery services unless we address four problems of scale. First, discovery services should monitor data access patterns to determine how best to replicate themselves, to determine whether to create specialized services that manage hot subsets of their data, and to diagnose accidentally looping clients. Second, discovery services should support significantly higher levels of replication, using algorithms specifically designed to function in the Internet's dynamic patchwork of autonomous systems. Third, the range of user expertise and needs requires that user interfaces and search strategies be highly customizable. Fourth, the Internet will need a hierarchically structured, extensible, object-caching service through which clients can retrieve data objects, once they are discovered. We consider these issues later.

SERVER INSTRUMENTATION

The designers of new information systems can never fully anticipate how their systems will be used. For this reason, we believe that new Internet services should instrument their query streams.

Self-instrumented servers could help determine where to place additional replicas of an entire service: If some X.500 server in Europe finds that half of its clients are in the U.S., the server itself could suggest that a strategically located replica be created.

Self-instrumented servers could also identify the access rate of items in their databases for use by more specialized services. For example, an instrumented Archie server would note that properly formed user queries only touch about 16% of Archie's database Such instrumentation would enable the creation of a complementary service that reported only popular, duplicate-free, or nearby objects. We discuss these ideas more later.

Self-instrumented servers could also identify server-client or server-server communication run amok. Large distributed systems suffer frequently from undiagnosed, endless cycles of requests. For example, self-instrumented Internet name servers show that DNS traffic consumes 20 times more bandwidth than it should, because of unanticipated interactions between clients and servers [6]. Self-instrumentation could identify problem specifications and implementations before they become widespread

and difficult to correct.

SERVER REPLICATION

Name servers scale well because their data are typically partitioned hierarchically. Because resource discovery tools search flat, rather than hierarchical or otherwise partitionable views of data, the only way to make these tools scale-is by replicating them. To gain an appreciation for the degree of replication required, consider the Archie file location service.

Currently, the global collection of Archie servers process approximately 100,000 queries per day, generated by a few thousand users worldwide. Every month or two of Internet growth requires yet another replica of Archie. Thirty Archie servers replicate a continuously evolving 150MB database of 2.1 million records. While a query posed on a Saturday night receives a response in seconds, it can take several hours to answer the same query on a Thursday afternoon. Even with no new Internet growth, for the current implementation of Archie to yield five-second response times during peak hours we would need at least 60 times more Archie servers, i.e., 1,800 servers. Because of its success and the continual rapid growth of the Internet, in time Archie will require thousands of replicas. Other successful tools that cannot partition their data easily will also require massive replication.
We believe massive replication requires additional research. On the one hand, without doubt we know how to replicate and cache data that partitions easily, as in the case of name servers [18]. Primary copy replication works well because name servers do not perform associative access, and organizational boundaries limit the size of a domain, allowing a handful of servers to meet performance demands.(4) We have learned many lessons to arrive at this understanding [6]. On the other hand, we have little experience deploying replication and caching algorithms to support massively replicated, flat, yet autonomously managed databases.

What do existing replication schemes for wide-area services lack? First, existing replication systems ignore network topology. They do not route their updates in a manner that makes efficient use of the Internet. One day, a streaming reliable multicast protocol might serve this purpose. Today, we believe, it is necessary to calculate the topology over which updates traverse and to manage replication groups that exploit the Internet's partitioning into autonomous domains.

Second, existing schemes do not guarantee timely and efficient updates in the face of frequent changes in physical topology, network partition, and temporary or permanent node failure. In essence, they treat all physical links as having equal bandwidth, delay, and reliability, and do not recognize administrative domains.
We believe that flooding-based replication algorithms can be extended to work well in the Internet. Both Archie and network news replicate using flooding algorithms. However, for lack of good tools, administrators of both Archie and network news manually configure the flooding topology over which updates travel, and reconfigure this topology manually when the physical network changes. This is not an easy task because Internet topology changes often, and manually composed maps are never

current While
we are developing tools to map the Internet [25], full network maps will not automate topology-update calculation.

Avoiding topology knowledge by using today's multicast protocols [8] for data distribution fails for other reasons. First, these protocols are limited to single routing domains or require manually placed tunnels between such domains. Second, Internet multicast attempts to minimize message delay, which is the wrong metric for bulk transport. At the very least, we see the need for different routing metrics. Third, more research is needed into reliable, bulk transport multicast that deals efficiently with site failure, network partition, and changes in the replication group.

We are exploring an approach to providing massively replicated, loosely consistent services [5]. This approach extends ideas presented in Lampson's Global name service [14] to operate in the Internet. Briefly, our approach organizes the replicas of a service into groups, imitating the idea behind the Internet's autonomous routing domains. Croup replicas estimate the physical topology and then create an update topology between the group members. The left-hand side of Figure 4 shows three replication domains.(Figure 4 omitted) Inside each replication domain, a logical update topology is established that best uses the physical topology connecting the replicas. Whenever the algorithm detects a meaningful change in physical topology among members of a replication domain, it modifies the logical update topology accordingly. Because it does not require a separate recovery algorithm, it is simpler than solutions based on Internet multicast.

DATA OBJECT CACHING

We believe that the Internet needs an object-caching service, through which search clients can retrieve the data objects they discover. We say this for two reasons.

First, people are using Mosaic, FTP and email as a cacheless, distributed file system [7]. Since the quantity and size of read-mostly data objects grows as we add new information services, an object cache would improve user response times and decrease network load (e.g., we found that one fifth of all NSFNET backbone traffic could be avoided by caching FTP data). Second, caches protect the network from looping client programs that repeatedly retrieve the same object. While the cache does not fix the faulty components, it does isolate the fault. A caching service would obviate the need for every new client and Internet service to implement its own data cache.

We believe the object cache should be hierarchically organized, as illustrated in Figure 5.(Figure 5 omitted) The dark circles in this figure represent file caches residing on secure machines placed near the entry points to regional networks and near the core of backbone networks. The organization of these caches could be similar to the organization of the Domain Name System. Clients would send their requests to one of their default cache servers. If the request missed the cache, the cache would resolve the request recursively with one of its parent caches, or directly from the FTP archive.

CLIENT CUSTOMIZATION

As the number of Internet information systems users gets larger, naturally it becomes more diverse. Allowing users flexible customization can be a great help, because people have different search styles and needs. To see how this can be done, consider the analogy of a newcomer to a town. One first establishes general acquaintance with the town layout and major services. One then learns about services close to one's heart -- for example, clubs, specialized stores, and recreation facilities-usually by word of mouth, the media, or advertising. After a while, one develops networks of friends, better knowledge of different services, and experience based on habits and interests. This is a continuing process, because new facilities appear, and one's interests evolve. The same process occurs on a much larger scale in the Internet and suggests ways that interactions between users and discovery services can be made flexible. Next we discuss three types of customization that can be addressed in discovery tools, based on some of our experimental systems.

The first type of customization involves tracking a person's search history. For example, recently one of this article's authors discovered that the New Republic magazine was available on-line while browsing Gopherspace via Mosaic. But only two days later it took him 15 minutes to navigate back to the same place. To address this problem, the user interface can keep track of previous successful and unsuccessful queries, comments a user made on past queries, and browsing paths. Existing systems record some of this information (e.g., the ability to set "bookmarks" in Gopher and "Hotlists" in Mosaic); saving the whole history can help further. For example, we built a system that records the paths users traverse when browsing FTP directories and allows this information to be searched [21]. It is also helpful to record search history and allow the searches themselves to be searched. For example, the X-Windows interface to agrep translates every command into the corresponding agrep statement, and allows flexible retrieval of this information. This type of history can support queries such as "What was the name of the service that allowed me to search for XYZ that I used about a year ago?"

The second type of customization is the ability to choose not only according to topic but also according to context and other attributes. If the search is for papers on induction, for example, knowledge of whether the searcher is a mathematician or an electrical engineer can be very useful. When one looks for a program named ZYX using Archie, for example, it would be useful to specify a preference for, say, a Unix implementation of the program.

The third type of customization is ranking. WAIS provides one form of ranking, in which matched documents are ordered by frequency of occurrence of the specified keywords. However, it would be useful to allow users to customize their notion of ranking, so they could choose to rank information by quality or reliability. Clearly, these notions are very subjective. If we are naive users, we may want information from someone who knows how to build easy-to-use systems. If we are academics, we may want information from someone with deep understanding. If we absolutely positively need it by tomorrow, we want someone who can deliver, and so on. We believe that in time the Internet will support many types of commercial and nonprofit

review services, and people will subscribe and follow recommendations from reviewers they trust (just like they go to movies or buy refrigerators based on reviews).

Customizations like the ones discussed here are missing from most current resource discovery client programs because they were not designed to be extensible.

DATA VOLUME

The amount of information freely available in the Internet is huge and growing rapidly. New usage modes will contribute additional scale. For example, as multimedia applications begin to proliferate, users will create and share voluminous audio, image, and video data, adding several orders of magnitude of data. Many users already store voluminous data, but do not share it over the Internet because of limited wide-area network bandwidths. For example, earth and space scientists collect sensor data at rates as high as gigabytes per day. To share data with colleagues, they send magnetic tapes through the postal mail. As Internet link capacities increase, more scientists will use the Internet to share data, adding several more orders of magnitude to the information space.

While resource discovery tools must deal with this growth, the scaling problems are not quite as bad as they may seem, because the number' and size of "searchable items" need not grow as fast. Resource discovery tools may be needed to find the existence and location of gigabytes or terabytes of raw sensor data, but probably they will not search or otherwise process all this data. A reasonably small-sized descriptor object will be sufficient to point anyone to this data. Only this descriptor object will need to be searched and indexed. The same holds for sound, video, and many other types of nontextual data. Searching image files is desirable, but current pattern-matching techniques are still too slow to allow large-scale

image processing on-the-fly. Again, a descriptor object can be associated with every image, describing it in words.

USER INTERACTION PARADIGMS: BROWSING VS. SEARCHING

Generally, there are two resource discovery paradigms in common use in the Internet-organizing/browsing and searching. Organizing refers to the human-guided process of deciding how to interrelate information, usually by placing it into some sort of a hierarchy (e.g., the hierarchy of directories in an FTP file system). Browsing refers to the corresponding human-guided activity of exploring the organization and contents of a resource space. Searching is a process in which the user provides some description of the resources being sought, and a discovery system locates information that matches the description.

We believe that a general discovery system will have to employ a combination of both paradigms -- here we analyze the strengths and weaknesses of each paradigm. The main weakness of organizing is that it is typically done by "someone else" and is not easy to change. For example the Library of Congress Classification System has Cavalry and Minor Services of Navies as two second-level topics (under "Military Science" and "Naval

Science" respectively), each equal to all of Mathematical Sciences (a second-level topic under "Science"), of which computer science is but a small part.(5) Ironically, this is also the main strength of organizing, because people prefer a fixed system they can get used to, even if it is not the most efficient.

Browsing suffers from this problem also because typically it depends heavily on the quality and relevance of the organization. Keeping a large amount of data well organized is difficult. In fact, the notion of "well organized" is highly subjective and personal. What one user finds clear and easy to browse may be difficult for users who have different needs or backgrounds. Browsing can lead to navigation problems also, and users can get disoriented. To some extent this problem can be alleviated by systems that support multiple views of information [19]. Yet, doing so really pushes the problem "up" a level -- users must locate appropriate views, which in itself is another discovery problem. Moreover, because there are few barriers to "publishing" information in the Internet (and we strongly believe there should not be any), there is great deal of information that is useful to only very few users, and often for only a short period of time. To other users, this information clutters the "information highway," making browsing difficult.

Searching is much more flexible and general than organizing/browsing, but it is also harder for the user. Forming good queries can be a difficult task, especially in an information space unfamiliar to the user. On the other hand, users are less prone to disorientation; the searching paradigm can handle change much better; and different services can be connected by searching more easily than by interfacing their organizations.

Many current systems, such as WAIS, Gopher, and World-Wide Web [1], employ an organization that is typically based on the location of the data, with limited per-item or perlocation searching facilities. Browsing is the first paradigm that users see,(6) but once a server or an archive is located, some type of searching is also provided. Searching can be comprehensive throughout the archive (for example, WAIS servers provide full-text indexes), or limited to titles.

INDEXING SCHEMES

The importance of searching can be seen by the recent emergence of file system search tools [11, 12, 16] and the Veronica system (a search facility for Gopher). Moreover, it is interesting to note that while the Prospero model [19] focuses on organizing and browsing, Prospero has found its most successful application as an interface to the Archie search system.

To support efficient searching, various means of indexing data are required. As illustrated in Figure 6, Internet indexing tools can be placed on a spectrum of indexing space vs. representativeness.(Figure 6 omitted) The upper-left corner of this figure is occupied by systems that achieve very space-efficient indexes, but only represent the names of the files or menus that they index. Archie and Veronica, for example, index the file and menu names of FTP and Gopher servers, respectively. The compact nature of these indexes means that a single index can support far-reaching searches. Yet, these indexes support very limited queries. For example, Archie

supports only name-based searches; searches based on content are possible only when the file names happen to reflect some of the contents.

The lower-right corner of Figure 6 is occupied by systems that provide full-text indexing of the data found at individual sites. For example, a WAIS index represents every keyword in a set of documents located at a single site. Similar indexes are available for individual Gopher and World-Wide Web servers. Some recent advances have lowered the space requirements for full-text indexing, with as low as 2% to 4% for the index used in Glimpse [16]. There is usually (but not always) a time-space trade-off; systems that use less space require more time for searching.

The middle region of Figure 6 is occupied by systems that represent some of the contents of the objects they index, based on selection procedures for including important keywords or excluding less meaningful keywords. For example, Essence and MIT's Semantic File System (SFS) select keys based on application-specific semantics of individual documents (e.g., locating the author lines within TeX documents). The Netfind site database includes keywords for each component of a site's Domain name, plus each keyword included in an organizational description that was constructed based on understanding the semantics of many information sources used to gather that information [22]. Whois++ [24] indexes templates that were manually constructed by site administrators wishing to describe the resources at their site.

By selecting relatively small but important information such as file names, Archie quickly became a popular search-based Internet resource discovery tool. But as we discussed earlier, Archie has scale problems. One way to overcome some of these problems is to introduce some hierarchy into the indexing mechanism in Archie (and similar tools). In addition to one flat database of all file names, it is possible to maintain much smaller slightly limited databases that will be replicated widely. For example, we can detect duplicates (exact and/or similar, see [151), and keep only one copy (e.g., based on locality) in the smaller databases. Most queries will be satisfied with the smaller databases, and only few of them will have to go further.

The scale problems for full texts are much more difficult. Full-text indexes are almost always inverted indexes, which store pointers to every occurrence of every word (possibly excluding some very common words). The main drawbacks of inverted indexes are their size -- usually 50% to 150% of the original text [10] -- and the time it takes to build and/or update them. Therefore, maintaining an inverted index of the whole Internet FTP space is probably out of the question. But separate local indexes, which is what we have now, do not present a general enough view of much of the available information. Often, users need to perform a lengthy browsing session to find the right indexes, and they often miss. This problem will only get worse with scale.

We envision a search system that connects local indexes and other pieces of information via a multilevel indexing scheme. The main principle behind such a scheme is that the number of search terms is quite limited no matter how much data exists. Search terms are mostly words, names, technical terms, and so forth, and the number of those is on the order of 10(6) to

10(7). But more important, this number grows more slowly than the general growth of information. Inverted indexes require enormous space because they catalog all occurrences of all terms. But if we index, for each term, only pointers to places that may be relevant, we end up with a much smaller index. Searching will be similar to browsing in the sense that the result of each query may be a list of suggestions for further exploration.

A big advantage of such a scheme is that the index can be partitioned in several ways, which makes it scalable. Specialized archives will, of course, keep their local indexes. Several local indexes can be-combined to form a two-level index, in which the top level can only filter queries to a subset of the local indexes. For example, separate collections of technical reports can form a combined index. Also, indexes can be combined according to topics or other shared attributes. The directory of services could be another index (but more widely replicated), which contains pointers to information about common terms and local information. There could be also a more detailed directory maintained at some servers with knowledge about more terms. Users could navigate by a combination of browsing and searching. In contrast with fixed browsing, this type of navigation will allow users to skip many levels, to customize their searches better, and to combine information from different sources more easily. Of course, many issues need to be resolved, including ranking, classification, replication, consistency, data extraction, privacy, and more.

An example (at a smaller scale) of the multilevel approach is Glimpse [16], an indexing and searching scheme designed for file systems. Glimpse divides the entire file system into blocks, and in contrast with inverted indexes, it stores for each word only the block numbers containing it. The index size is typically only 2% to 4% of the text size (hence its position in Figure 6). The search is done first on the index and then on the blocks that match the query. Glimpse supports approximate matching, regular expression matching, and many other options.

Essence and the MIT Semantic File System do selective indexing of documents, by selecting keywords based on knowledge of the structure of the documents being indexed. For example, Essence understands the structure of several common word processing systems (as well as most other common file types in Unix environments), and uses this understanding to extract authors, titles, and abstracts from text documents. In this way, it is able to select fewer keywords for the index, yet retain many of the keywords that users would likely use to locate documents. Because these types of systems exploit knowledge of the structure of the documents being indexed, it would be possible to include document structure information in the index. (This idea was discussed earlier.)

TOPIC SPECIALIZATION, CLASSIFICATION, AND INFORMATION QUALITY

It is safe to say that at least 99% of the available data is of no interest to at least 99% of the users. The obvious solution to this problem is to construct specialized archives for particular domains of interest. We believe new techniques are needed to simplify the process of managing specialized archives.

Currently, specialized archives rely on 1) direct contributions from their

communities and 2) administrators who contribute time and expertise to keep the collections well structured. Except for replication (or mirrors as they are usually called in the Internet) of FTP files, archives do not cooperate among themselves.

We see the need for a discovery architecture and

set of tools that easily
let people create specialized services and that automatically discover information from other services, summarize it, keep it consistent, and present it to the archive administrator for their editorial decision. An important component of any complete solution is a directory of services in which archives describe their interest specialization and keep this definition current.

This approach amounts essentially to defining archives in terms of queries [2, 5]. A server uses its query periodically to hunt throughout the Internet for relevant data objects. One type of server could, for example, be specialized to scan FTP archives, summarizing files and making these summaries available to yet other services.

Such an architecture could greatly reduce the manual steps that archive administrators perform to incorporate users' contributions. This architecture would also help users discover smaller, highly specialized archives.

To help support topic-specialized servers, we believe information must be classified also according to topic or community-specific taxonomies. For example, an archive of computer science technical reports might be classified using the ACM Computing Reviews taxonomy. Taxonomies allow a more uniform search space than is possible solely by content-indexing documents. Because a particular document may be of value to several communities, it should be possible to classify documents according to multiple taxonomies, and register classification information for the document in several specialized archives.

Classification should also include some description of information quality. Often, for example, scientists need to understand such things as the methods by which data were gathered and what error controls were used. At this point it is not clear how to specify quality, because there are many associated issues. At a minimum it would be useful to record the author, data collection process, and something about the review process to which the data were subjected before being published. Other possibilities might include such things as pointers to conflicting points of view and support for "voting" by users who have perused the data.

We believe tools should be developed that allow authors to mark up their documents with classification terms from some selected set of taxonomies. In turn these classification data should be included in topic indexes, with attributes indicating that the source of the information was a classification process, rather than just a content-indexing process (since the latter tends to generate many less well-focused or well-defined terms).

SUMMARY

The Internet's rapidly growing data volume, user base, and data diversity will create difficult problems for the current set of resource discovery tools. Future tools must scale with the diversity of information systems, number of users, and size of the information space.

With growing information diversity, techniques are needed to gather data from heterogeneous sources and sort through the inherent inconsistency and incompleteness. Internet Research Task Force efforts in this realm focus on application-specific means of extracting and cross-correlating information, based on both explicit and implicit data-typing schemes.

With a growing user base, significantly more load will be placed on Internet links and servers. This load will require much more heavily replicated servers and more significant use of data caching, highly customizable client interfaces, and self-instrumenting servers to help guide replication and specialization. IRTF efforts in this realm focus on flooding-based replication algorithms that adapt to topology changes, and on customized clients.

As the volume of information continues to grow, organizing and browsing data break down as primary means for supporting resource discovery. At this scale, discovery systems will need to support scalable content-based search mechanisms. Current systems tend to strike a compromise between index representativeness and space efficiency. Future systems will need to support indexes that are both representative and space efficient. IRTF efforts in this realm focus on scalable content-based searching algorithms, and on servers specialized to support particular user communities.

FOOTNOTES

(1) The reader interested in an overview of resource discovery systems and their approaches is referred to [20].

(2) FTP is an Internet standard protocol that supports transferring files between interconnected hosts. Anonymous FTP is a convention for allowing Internet users to transfer files to and from machines on which they do not have accounts, for example, to support distribution of public domain software.

(3) Efforts are under way to improve this gateway.

(4) Because it is both a name service and a discovery tool, X.500 could benefit from massive replication.

(5) There was a time, of course, when the study of cavalry was much more important than the study of computers.

(6) WAIS supports searching at the top level via the directory of servers. but many users simply browse a local copy of this server list.

REFERENCES

1. Berners-Lee, T., Cailliau, R., Groff, J.F., and Pollermann, B. World-Wide Web: The information universe. Elec. Netw. Res. Appl. Pol. 1,2

(Spring 1992), 52-58.

2. Bowman, C.M. and Dharap, C. The Enterprise distributed white-pages service. In Proceedings of the USENIX Winter Conference. USENIX Association, Berkeley, Calif., 1993. pp. 349-360.

3. Bowman, M., Dharap, C., Baruah, M., Camargo, B., and Potti, S. A file system for information management. In Proceedings of the Conference on Intelligent Information Management Systems (Washington, D.C., June 1994).

4. Bowman, M., Peterson, L.L., and Yeatts, A. Univers: An attribute-based name server. Softw. Prac. Exp. 20, 4 (Apr. 1990), 403-424.

5. Danzig, P.B., Li, S.-H., and Obraczka, K. Distributed indexing of autonomous Internet services. Comput Syst. 5, 4 (Fall 1992), 433-459.

6. Danzig, P.B., Obraczka, K., and Kumar, A. An analysis of wide-area name server traffic: A study of the Domain Name System. In ACM SIGCOMM '92 Conferenee. ACM, New York, 1992, pp. 281-292.

7. Danzig, P.B., Schwartz, M., and Hall, R. A case for caching file objects inside internetworks. In ACM SIGCOMM '93 Conference. ACM, New York, 1993, pp. 239-248.

8. Deering, S. and Cheriton, D. Multicast routing in datagram internetworks and extended LANs. ACM Trans. Comput. Syst. 8, 2 (May 1990), 85-110.

9. Emtage, A. and Deutsch, P. Archie: An electronic directory service for the Internet. In Proceedings of the Winter 1992 Usenix Conference. ACM, New York, 1992, pp. 93-110.

10. Faloutsos, C. Access methods for text. ACM Comput. Surv. 17 (Mar. 1985), 49-74.

11. Gifford, D.K., Jouvelot, P., Sheldon, M.A., and O'Toole, J.W., Jr. Semantic file systems. In Proceedings of the Thirteenth ACM Symposium on Operating Systems principles. ACM, New York, 1991, pp. 16-25.

12. Hardy, D. and Schwartz, M.F. Essence: A resource discovery system based on semantic file indexing. In Proceedings of the USENIX Winter Conference. USENIX Association, Berkeley, Calif., 1993, pp. 361-374.

13. Khale, B. and Medlar, A. An information system for corporate users: Wide Area Information Servers. ConneXions Interoper. Rep. 5, 11(Nov. 1991), 2-9.

14. Lampson, B. Designing a global name service. In Proceedings of ACM Principles of Distributed Computing. ACM, New York, 1986, pp. 1-10.

15. Manber, U. Finding similar files in a large file system. In Proceedings of the USENIX Winter Conference. USENIX Association, Berkeley, CaliE, 1994, pp. 1-10.

16. Manber, U. and Wu, S. Glimpse: A tool to search through entire file systems. In Proceedings of the USENIX Winter Conferenee. ACM, New York,

1994, pp. 23-32.

17. McCahill, M. The Internet Gopher: A distributed server information system. ConneXions Interoper. Rep. 6, 7 (July 1992), 10-14.

18. Mockapetris, P. RFC 1034: Domain names -- concepts and facilities. Internet Req. Comm. (Nov. 1987).

19. Neuman, B.C. Prospero: A tool for organizing Internet resources. Elec. Netw. Res. Appl. Pol. 2, 1 (Spring 1992), 30-37.

20. Schwartz, M.F., Emtage, A., Kahle, B., and Neuman, B.C. A comparison of Internet resource discovery approaches. Comput. Syst. 5, 4 (Fall 1992), 461-493.

21. Schwartz, M.F., Hardy, D.R., Heinzman, W.K., and Hirschowitz, G. Supporting resource discovery among public internet archives using a spectrum of information quality. In Proceedings of the Eleventh International Conference on Distributed Computing Systems. 1991, 82-89.

22. Schwartz, M.F. and Calton, P. Applying an information gathering architecture to Netfind: A White Pages tool for a changing and growing Internet. IEEE/ACM Trans. Netw. To be published.

23. Sun Microsystems. SunOS Reference Manual. Sun Microsystems, Mountain View, CA, 1990.

24. Weider, C., Fullton, J., and Spero, S. Architecture of the Whois++ index service. Tech. Rep., Nov. 1992. Available by anonymous FTP from nri.reston.va.us, in internet-drafts/draft-ietfwnils-whois-00.txt.

25. Wood, D.C.M., Coleman, S.S., and Schwartz, M.F. Fremont: A system for discovering network characteristics and problems. In Proceedings of the USENIX Winter Conference. USENIX Association, Berkeley, Calif., pp. 335-348.

About the Authors:

C. Mic Bowman is a member of technical staff at Transarc Corporation. Current research interests include descriptive naming systems for local and wide-area file systems, structured file systems, resource discovery, and protocols for remote computing. Author's Present'Address: Transarc Corp., The Gulf Tower, 707 Grant Street, Pittsburgh, PA 15219; email: mic(at)transarc.com

Peter B. Danzig is an assistant professor of computer science at the University of Southern California. Current research interests include the measurement and performance debugging of Internet services, distributed system architectures for resource discovery, and mathematical modeling of communication networks. Author's Present Address: Computer Science Department, University of Southern California, 941 W. 37th Place, Los Angeles, CA 90089-0781; email: danzig(at)usc.edu

Udi Manber is a professor of computer science at the University of Arizona. Current research interests include design of algorithms, pattern matching,

computer networks, and software tools. Author's Present Address: Computer Science Department, University of Arizona, Tucson, AZ 85721; email: udi(at)cs.arizona.edu

Michael F. Schwartz is an associate professor of computer science at the University of Colorado. Current research interests include issues raised by international networks and distributed systems, with particular focus on resource discovery and wide-area network measurement. Author's Present Address: Computer Science Department, University of Colorado, Boulder, CO 80309-0430; email: schwartz(at)cs.colorado.edu

Bowman is supported in part by the National Science Foundation under grants CDA.8914587 and CDA-8914587AO2, the Advanced Research Projects Agency under contract number DABT63-93-C-0052, and an equipment grant from Sun Microsystems, Inc. Danzig is supported in part by the Air Force Office of Scientific Research under Award Number F49620-93-1-0082, by a National Science Foundation smallscale infrastructure grant CDA-9216321, and by the Advanced Research Projects Agency under contract number DABT63-93-C-0052. Manber is supported in part by the National Science Foundation under grant numbers CCR-9002351 and CCR-9301129, and by the Advanced Research Projects Agency under contract number DABT63-93-C-0052. Schwartz is supported in part by the National Science Foundation under grant numbers NCR-9105372 and NCR-9204853, the Advanced Research Projects Agency under contract number DABT63-93-C-0052, and an equipment grant from Sun Microsystems' Collaborative Research Program.

The information contained in this article does not necessarily reflect the position or the policy of the U.S. government or other sponsors of this research. No official endorsement should be inferred.

THIS IS THE FULL-TEXT.

Geographic Names: US

Descriptors: Internet; Communications networks; Information retrieval; Growth rate; Problems; Protocol; Studies
Classification Codes: 9190 (CN=United States); 5250 (CN=Telecommunications systems); 9130 (CN=Experimental/Theoretical)

HIERARCHICAL DIRECTORIES ON CMS MINIDISKS

A hierarchical file organization improves CMS file management and organization. It can also reduce the need for multiple minidisks. This disclosure describes how to support hierarchical directories on CMS minidisks while maintaining compatibility with the current CMS file system structure and existing CMS programs.

The flat CMS file system makes file organization very difficult. In order to organize files into meaningful parts, the CMS user must use multiple minidisks. But, minidisk management is even more complicated since minidisks are fixed in size and users can only access up to 26 minidisks at any one time. Finally, because the CMS limits file names and types to eight characters, it is difficult to group files with meaningful names.

This article describes a technique which extends the CMS file system to support a hierarchical organization. There are three major parts:

1. The structure on external storage (the minidisk) is basically unchanged. Traditionally, the directory for a CMS minidisk is a standard fixed-format, CMS file which only the CMS file system manipulates. The user is not aware of its existence. With hierarchical directories, a minidisk can contain many directory files that are still managed only by new or modified CMS file system code. Now, the user can see these directory files and manipulate them. "The Real Directory Tree", described later, is a sample hierarchical directory on a minidisk. Directory management is very simple. For most functions, such as erase and rename, directory files are treated the same as any other CMS file.

2. The structure in CMS virtual storage ("The Virtual Directory Tree" and "The Virtual Storage Structure" being described later) has an expanded control block structure, called Shadow ADTs, which add a second dimension to the file directory structure. The Shadow ADT concept is key to this disclosure.

   The Active Device Table (ADT) is a CMS control block which defines each accessed minidisk. Each ADT is assigned a mode letter from "A" to "Z" by the user, which limits CMS to 26 active minidisks at any one time. Shadow ADTs allow an unlimited number of ADTs to be associated with each mode letter and to represent an unlimited number of directories from the same minidisk. The new structure still has an ADT which represents both the minidisk and the root directory as accessed by mode letter. From the root ADT, there is

now a chain of Shadow ADTs -- one for each additional directory.

The relation and structure among different CMS mode letters is still the same. The new structure only affects the organization within a mode letter (a minidisk). During a search for a file on a minidisk, CMS traverses the Shadow ADT chain in the order defined by the Path function.

3. Using the new control block structure in storage as a base, a set of new functions and techniques help CMS manage hierarchical directories (see "The Virtual Storage Structure"). These are:

   PATH associates one or more directories with a mode letter. The first directory (the root) is always associated with a mode letter by the CMS ACCESS command. All or a subset of the sub-directories of the accessed directory can be added to the mode with the Path function.

   All ADTs are now doubly linked with a set of static pointers to the next and previous ADT in the list. These pointers represent the path-defined order of the directories and remain constant during the session.

   SEARCH increases the flexibility of hierarchical directories without losing the traditional CMS search order. The Path function defines a default search order for directories. Essentially, pathing defines a two-dimensional search order. CMS file operations (e.g., STATE) continue to search each mode in alphabetical order but also search all directories for each mode in the path-defined order.

   INVISIBLE directories are a technique used to manage intervening directories in a real tree that are not included in the path associated with a mode letter. In order to use any directory in a tree, CMS creates Shadow ADTs for all directories in the intervening path. Directories of a real tree that are not specified on the path are invisible to the user.

   CURRENT defines which of the directories in the path is the first directory searched and the only directory for output. There is one current directory per mode letter. The Current function can make any of the directories in the path the current directory.

   CMS uses a set of dynamic pointers to define the current order in which to search directories. CMS can use the static pointers to return the search order to that originally defined by the Path function.

   PUSH/PULL is a technique to allow easy and rapid traversing of directories along a path of a tree. The current directory is pushed onto a stack, and the new sub-directory being viewed

becomes the current directory. The dynamic pointers in the Shadow ADTs are manipulated as the user traverses the tree. Pull then restores the pushed directory from the stack as the user returns.

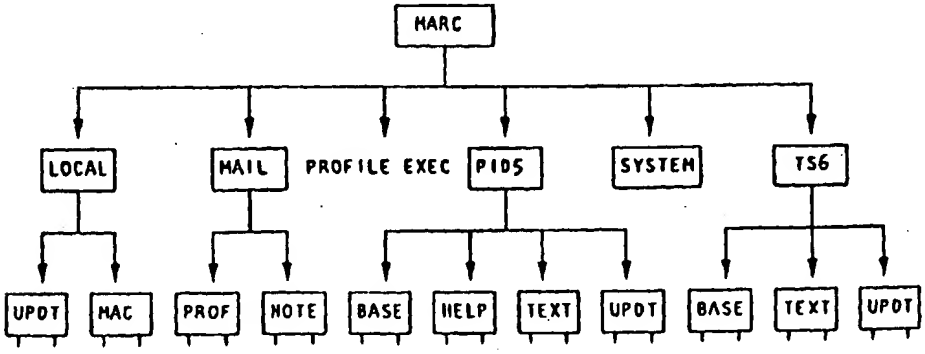
COMMIT of changes to disk starts at leaf nodes and propagates any changes through all parents to the root directory. CMS starts from each leaf node and traverses the tree to the root, using the parent pointer in each Shadow ADT. Whenever CMS detects that it has already traversed a particular portion of the tree, it skips to the next leaf node. This avoids redundant traverses of any part of a path that is common to more than one leaf node.

CMS uses the root directory, to which all directories have a pointer, to maintain information common to all directories in the tree.

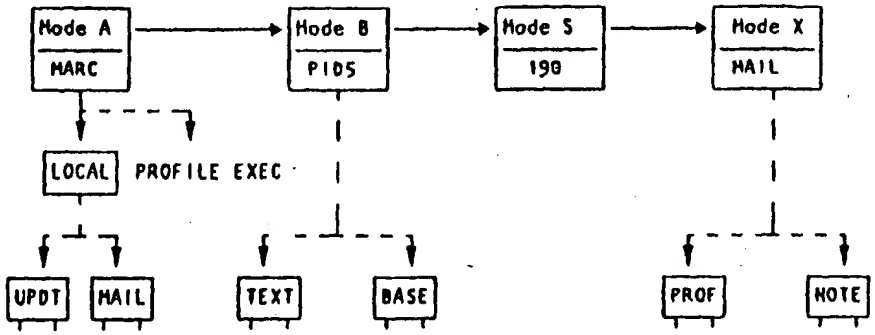
QUERY provides the names of the directories associated with a mode letter in either path-defined or search-defined order. Query also gives the names of the parent, current and root directories.

The hierarchical file organization does not affect existing applications. Users can still have a flat directory structure by having only one directory per mode letter, or they can build a hierarchy. Because there is no limit to the depth of the hierarchy, users can organize files into smaller groups with meaningful names. Additionally, there is an 8-character name for each level in the hierarchy which provides for fully qualified names. Duplicate CMS file names and types can exist in separate directories.

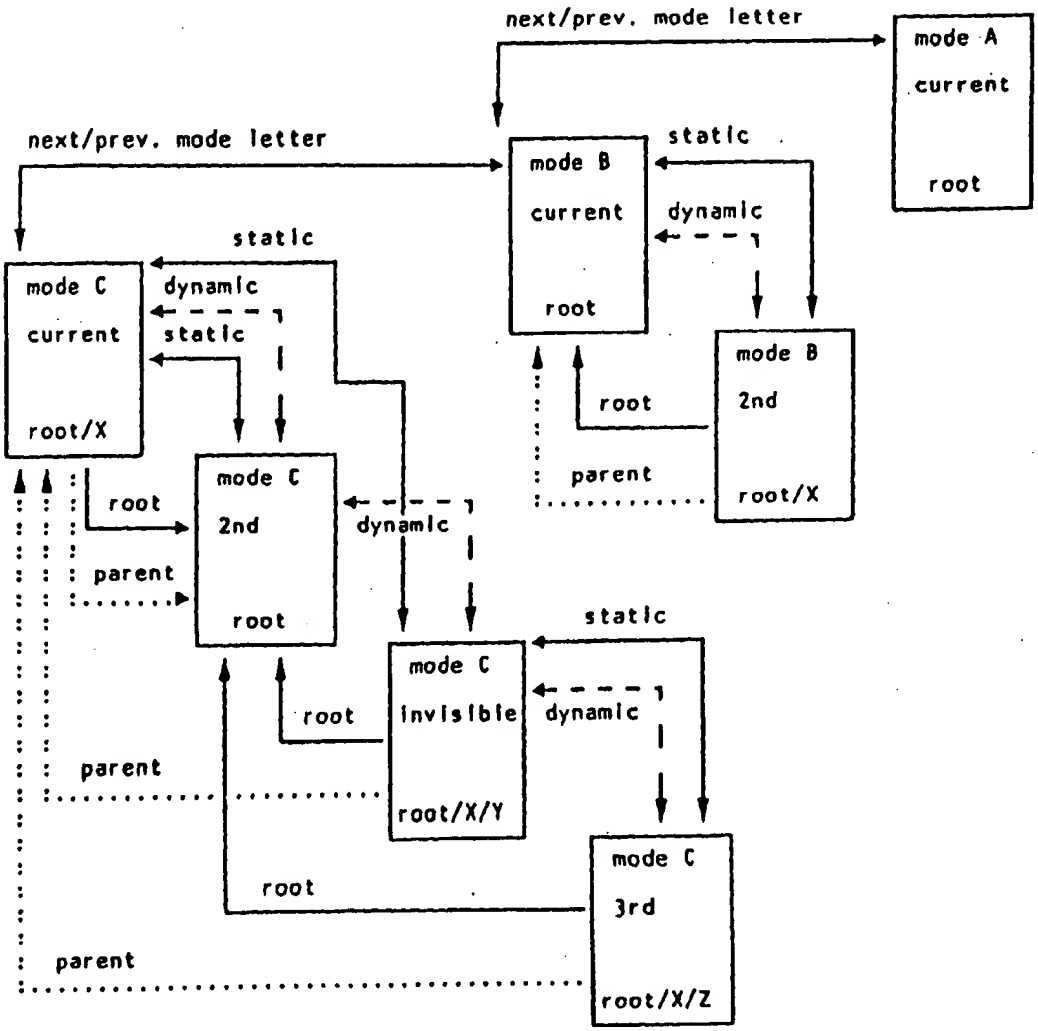
THE REAL DIRECTORY TREE below is a sample hierarchical directory on a minidisk. The boxes represent directories and the arrows represent the real path among the nodes. In the example, MARC is the name of the root directory and is the parent of the directories called LOCAL, MAIL, PIDS5, SYSTEM, and TS6. PROFILE EXEC is a file in the directory named MARC. The UPDT directory is a leaf directory, and the path to it is MARC/LOCAL/UPDT.



THE VIRTUAL DIRECTORY TREE is a sample hierarchical directory that a user might create from the above hierarchy in virtual storage. The current directory for each mode letter is MARC, PID5, 190 (which has no hierarchy defined), and MAIL. The path and search order for the directory MAIL is MARC/LOCAL/MAIL which is different from the real path shown above.



THE VIRTUAL STORAGE STRUCTURE shown on the following page is a detailed sample of the control block relationships for hierarchical directories in virtual storage. Mode letter "A" has a single directory associated with it. Mode letter "B" has the real root directory as the current directory and a directory named "X" second in the search order. Mode letter "C" has several directories in the path starting with directory "X" as the current directory. The static pointers reflect the original path-defined order. The dynamic pointers reflect the search-defined order. All directories point to both the root directory and their parent directory. Directory "root/X/Y" is invisible since it is not on the path defined by the user and will be skipped during directory search processes.


next/prev. mode letter
mode A
current
root
next/prev. mode letter
mode B
current
root
static
dynamic
mode C
current
root/X
static
dynamic
static
root
mode B
2nd
root/X
root
parent
root
mode C
2nd
root
dynamic
parent
static
mode C
invisible
root/X/Y
root
dynamic
parent
mode C
3rd
root/X/Z
root
parent

April 1983 ... IBM TECHNICAL DISCLOSURE BULLETIN ... <-- pp5911-5913 -->

Title:

Virtual File Access to a Binary Relational Database.

Index terms: ...formation Retrieval Addressing JEA
Text

TDB

This article describes a mechanism for obtaining access to a binary relational database via program operations which are equivalent to record input/output operations on a conventional file. This mechanism allows a simple program to load a conventional file into a binary relational database, or unload a part of a database into a file. It also permits the simple migration of existing application programs, which access files, to operate on a database.

Query operations on a binary relational database have been described in (1,2). A query is defined as a regular, structured collection of data items extracted from a database, and is therefore analogous to a conventional file of data records. In this article, a query is redefined as a "virtual file" which may be materialized from a data-base, data-base, although it does not exist as a physical entity within the data-base. Alternatively, records from an external source may be inserted into a virtual file, resulting in update operations on the database, although the external record is not stored as a physical entity.

Before a program can access a virtual file, some preparatory actions must be performed:

```
A.  Define a query over the database by one of the known
methods.
B.  Associate with each column in the query a field name, field
width and field offset within a virtual record.
C.  Store the description of the query and the description of
each virtual field (i.e., the results of steps A and B) in a
data dictionary.
D.  Associate a virtual file name with the query, and store this
in the data dictionary.
```

Step C has the effect of linking a description of a source of data (the query) with its intended destination (the output record). Step D provides a name which may be used in an application program to refer to the virtual file so defined. Existing methods do not permit these associations to be made.

The data dictionary which is used in steps C and D should be the same data dictionary which is used to contain the description of the binary relational database structure. It is then possible to record the description of the required query as a search path which refers to specified database sets and database relationships.

The major program operations on a virtual file and the methods by which these can be implemented are:

```
1.  OPEN virtual file name
2.  CLOSE virtual file name
3.  GET virtual-record FROM virtual file name
4.  PUT virtual-record INTO virtual file name
5.  DELETE virtual-record FROM virtual-file-name
6.  INSERT virtual-record INTO virtual-file name.
```

1. The OPEN operation searches the data dictionary for the specified virtual file name, and returns an error indication if it is not found. Otherwise, it retrieves the query definition

from the data dictionary and holds it in main storage.

2. CLOSE searches main storage for the virtual file name and returns an error indication if it is not found. Otherwise, it releases the storage used to hold the query definition and the virtual file name.

3. GET, PUT, DELETE, and INSERT also test whether OPEN has been performed previously. GET executes the query definition in main storage, so as to retrieve a single row of the query output. It places this row, field by field, into the virtual record receiver and retains an indication of the last row retrieved.

4. PUT receives a virtual-record containing updated fields, and replaces it into the database. It does this by re-executing the query definition and comparing the original contents of the database with the record received. Each field which differs from the original is placed into the appropriate database set and linked with the element corresponding to the record key via the appropriate database relationship. The corresponding link to the original field is severed.

5. DELETE erases the record just retrieved from the database. It does this by locating the element in the database corresponding to the record key, then deleting this element from the database and severing all its links to other fields.

6. INSERT receives a virtual-record and places it into the data-base. It does this by placing each field in the virtual record into the appropriate database set if it is not already a member of that set, then inserting the record key field into its appropriate database set, and linking it with each of the field elements just placed in the data-base via the appropriate database relationship.

The advantage of this method is that each of the operations 1 to 6 may be defined generically for all possible virtual files. Each operation interprets the information obtained from the data dictionary to determine the specific record form and contents for that operation. The application program is therefore bound only to the virtual record format described, and is not bound to the query which was executed to generate this virtual record. The query may subsequently be changed providing the record format is still honored.

Thus, the application program has a high degree of data independence.

References

(1) D.J. Pullin and G. C. H. Sharman, "Model Query Generation," IBM Technical Disclosure Bulletin 25, 11A, 5499-5500 (April 1983).

(2) D.J. Pullin and G. C. H. Sharman, "Method for Modifying Database Queries," IBM Technical Disclosure Bulletin 25, 11A, 5501-5502 (April 1983).

Diagrams: one
Order/Fcode/Docket: 83A 60811 / 83-000, 54-000. P01 / UK8810092
April 1983 ... IBM TECHNICAL DISCLOSURE BULLETIN ... <-- pp5911-5913 -->

055; 12-31-2007"
@PJL SET JOBATTR="JobAcct1=ecolbert1"
@PJL SET JOBATTR="JobAcct2=

5589644/19 **Links**
JAPIO

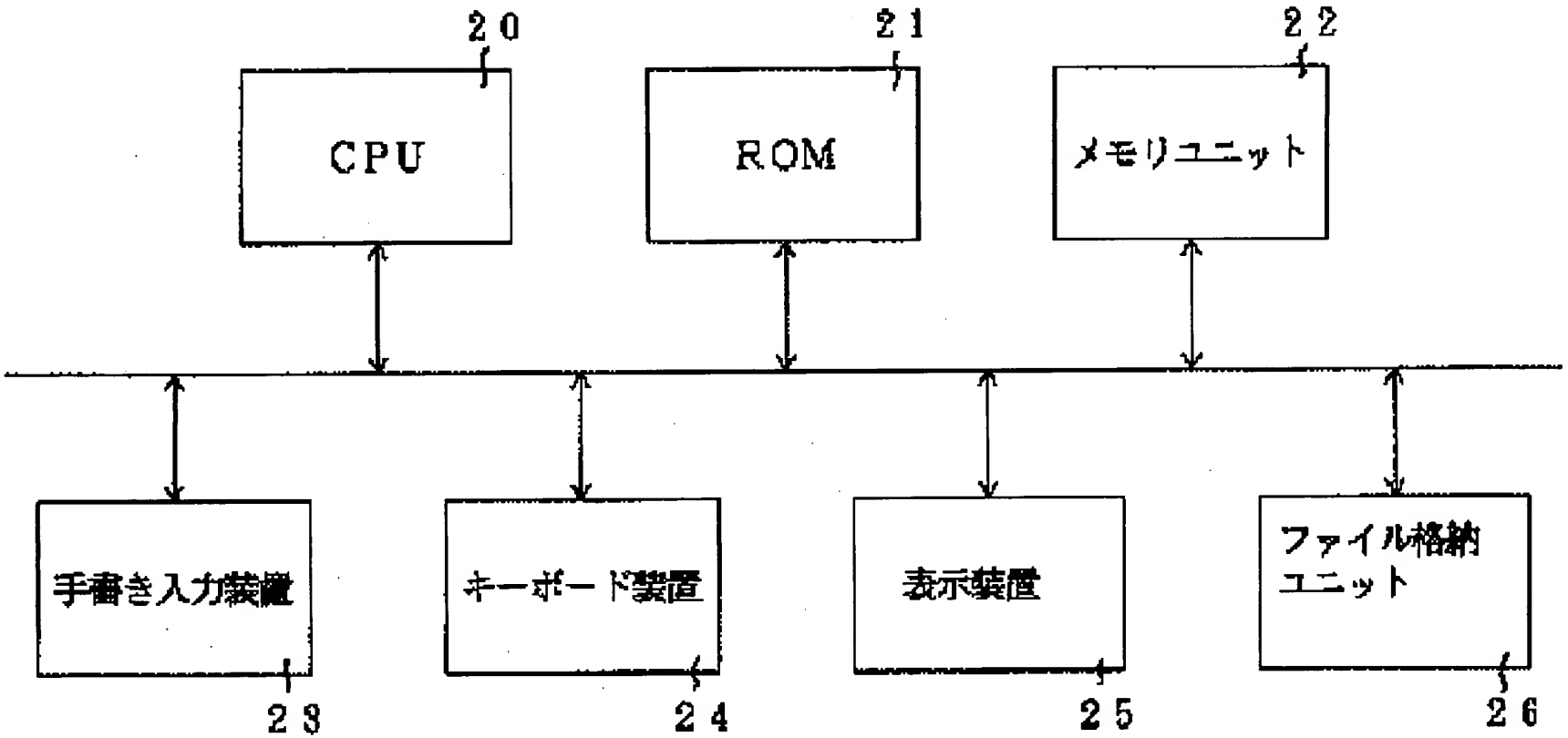
05589644 **Image available**

INFORMATION PROCESSING SYSTEM AND RECORDING MEDIUM STORING PROGRAM FOR COMPUTER TO PERFORM PROCESSING FOR THE SAME

Pub. No.: 09-204444 [JP 9204444 A]
Published: August 05, 1997 (19970805)
Inventor: OKAWA TORU
SATO YASUO
MATSUKURA RYUICHI
Applicant: FUJITSU LTD [000522] (A Japanese Company or Corporation), JP (Japan)
Application No.: 08-306871 [JP 96306871]
Filed: November 18, 1996 (19961118)
International Class: [6] G06F-017/30; G06F-012/00; G06T-001/00
JAPIO Class: 45.4 (INFORMATION PROCESSING -- Computer Applications); 45.2 (INFORMATION PROCESSING -- Memory Units); 45.9 (INFORMATION PROCESSING -- Other)
JAPIO Keyword: R011 (LIQUID CRYSTALS)

ABSTRACT
PROBLEM TO BE SOLVED: To enable retrieval processing based on non-coded data such as image data or audio data by preparing label information from non-coded information at one part of a document and performing retrieval processing, etc., based on the relation of correspondence between the label information and the document.

SOLUTION: A CPU 20 extracts features from image data, the number of blocks is extracted as the feature and the number of blocks is generated as the label information corresponding to the image data document. A table showing the relation of the image data document and a storage position is constructed inside a file storage unit 26, and the relation of the both is managed inside a system. Then, the label information is prepared and described in the label managing table so as to correspond to the prepared and edited image data document and while using this table, the relation of the image data document and the label information is managed. Further, retrieval data are generated, the retrieval data are compared with the read label image information and when they are coincident, these data are set to a prescribed register as the ID of label information.


20
21
22
CPU
ROM
メモリユニット
手書き入力装置
キーボード装置
表示装置
ファイル格納
ユニット
23
24
25
26